基于交通大数据的公交排班和调度机制研究

Bus Scheduling and Dispatching Mechanism Research Based on Traffic Big Data

学生姓名: 谢嘉昊

学 号: 20309082

院 系: 电子与信息工程学院

专 业: 通信工程

指导教师: 王玺钧 副教授

时间：二〇二四年四月二十八日

学术诚信声明

本人郑重声明：所呈交的毕业论文（设计），是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文（设计）不包含任何其他个人或集体已经发表或撰写过的作品成果。对本论文（设计）的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本论文（设计）的知识产权归属于培养单位。本人完全意识到本声明的法律结果由本人承担。

作者签名:

日　　期：　　年　　月　　日

论文题目： 基于交通大数据的公交排班和调度机制研究
专　　业： 通信工程
学生姓名： 谢嘉昊
学　　号： 20309082
指导教师： 王玺钧 副教授

摘要

基于交通大数据进行交通规划是当前交通领域的研究热点之一，而公交运营中的时刻表排班和公交车辆调度问题是交通规划中的重要一环。

针对公交排班调度问题中的时刻表排班，本文提出了一种基于深度强化学习的双向动态公交时刻表排班算法（Deep Reinforcement Learning-based dynamic bus Timetable Scheduling method with Bidirectional Constraints，DRL-TSBC），将双向的公交时刻表排班问题建模成马尔可夫决策过程，使用深度 Q 网络决定是否进行发车。经过测试，DRL-TSBC 在有效减少乘客的平均等待时间的同时，实现了上下行发车次数相等。此外，在客流量出现突变时，DRL-TSBC 可以根据变化的客流做出实时调整。

另一个重要的问题是公交车辆调度问题。针对此问题，本文提出了一种基于深度强化学习的公交车辆调度算法（Deep Reinforcement Learning-based Bus Scheduling Algorithm，DRL-BSA），将该问题建模成马尔可夫决策过程，使用竞争深度双 Q 网络，实现使用最少的车辆数量完成发车计划，并尽量使得所有车辆的发车次数均为偶数次。当出现突发的交通拥堵时，DRL-BSA 可以根据情况进行实时调整，在不增加使用车辆数量的情况下调整发车计划。

关键词： 交通大数据; 公交排班问题; 深度强化学习

Title: Bus Scheduling and Dispatching Mechanism Research Based on Traffic Big Data
Major: Communication Engineering
Name: Xie Jiahao
Student ID: 20309082
Supervisor: Assoc. Prof. Wang Xijun

ABSTRACT

Conducting traffic planning based on traffic big data is one of the current research hotspots in the field of transportation, and the planning research in bus operations is an important part of traffic planning.

This paper propose a Deep Reinforcement Learning-based dynamic bus Timetable Scheduling method with Bidirectional Constraints (DRL-TSBC) for the timetable scheduling problem. In DRL-TSBC, the bidirectional bus timetable scheduling problem is modeled as a Markov decision process, and a Deep Q Network (DQN) is used to make decisions on whether to depart buses. After testing, DRL-TSBC effectively reduces the average waiting time for passengers while achieving an equal number of departures for both directions. Moreover, when there is a sudden change in passenger flow, DRL-TSBC can make real-time adjustments according to the changing passenger flow.

Another important issue is the bus vehicle scheduling problem. Addressing this, the paper propose a Deep Reinforcement Learning-based Bus Scheduling Algorithm (DRL-BSA), which models the problem as a Markov decision process and uses a Dueling Double Deep Q Network (D3QN) to select vehicles at each departure point that needs to depart, achieving the use of the minimum number of vehicles to complete the departure plan, and trying to ensure that the number of departures for all vehicles is an even number. When sudden traffic congestion occurs, DRL-BSA can make real-time adjustments according to the situation, adjusting the schedule without increasing the number of vehicles used.

Keywords: Traffic big data, Bus scheduling problem, Deep Reinforcement Learning

目录

第 1 章 引言

1.1 选题背景与意义

目前，由于人们日益增长的出行需要，城市中的私家车数量在不断增加，大量的私家车导致了大城市中频繁出现的交通拥堵问题，同时也导致了能源消耗、环境污染等问题。随着这些问题的不断加剧，城市交通的可持续发展成为了一个愈发重要的议题[1]。在城市交通系统中，公共交通系统是一个至关重要的组成部分，其包括公交系统、地铁系统和出租车系统等。一个发达且便捷的公共交通系统，可以减少能源损耗、保护环境，并显著减轻交通拥堵问题。但是由于在当前的公共交通系统中，存在等待时间过长和过于拥挤等问题，许多人在出行的时候并不会选择使用公共交通工具[2]。于此同时，由于公共交通系统的规划往往不是最优的，乘客乘坐公共交通工具出行所需的时间会出现不必要的增加，造成了严重的经济浪费[3]。所以，提升公共交通系统的效率是非常有必要的。公交系统又是公共交通系统中的一个高客运量的重要组成部分，提升公交服务质量也是城市公共交通规划问题中的重点。通过优化公交系统，可以在很大程度上提升乘客对于公共交通系统的满意程度，同时降低公交公司的运营成本。

在公交系统中，公交排班调度问题一般分为两个部分进行处理，首先生成公交发车时刻表，即公交时刻表排班，随后基于这个时刻表安排车辆，即公交车辆调度。对公交时刻表进行设计是一个非常重要的工作流程，其关系到公交车辆的运行效率与乘客的出行体验。通过优化公交时刻表，可以缩短乘客的平均等待时间，减少公交车辆的空载率，提高公交系统的运行效率。有效的公交发车时刻表应在发车次数和乘客等待时间之间取得平衡，以满足公交公司和乘客的不同需求[4]。此外，大多数公交线路上，同一条线路的两个方向具有相同的首末站点。为确保公交车在一天结束时返回原出发站点，这两个方向的总发车次数需保持一致。若上下行方向的发车次数不等，则会出现车辆需要空车返回原出发站点的情况，造成一定的浪费。然而，由于乘客的出行需求是动态变化的，易受无法提前预知的情况的影响，例如大型活动、天气原因等，导致预先设计好的公交时刻表无法很好地适应出现变化后的客流量。因此，通过交通大数据的支撑，使用创新算法设计出能够动态调整的公交时刻表，并保证上下行发车次数相等是非常重要的。

在完成了公交时刻表排班后，另外一个重要的工作流程是按照该公交时刻表安排车辆，即进行公交车辆调度。如何合理地安排车辆以节省成本是公交公司十

分重视的问题[5]，而对于每一辆车来说，由于公交司机一般会居住在首发站附近，在一天的工作结束后司机需要返回原出发站点，这就要求每一辆车辆完成的行程次数应该为偶数次。在实际的运营过程中，由于交通事故、天气原因等问题，车辆可能无法按照原计划完成任务，此时需要对车辆进行重新安排。因此，以交通大数据为立足点，设计出能够动态调整、减少开销的公交车辆调度方案同样是非常重要的。

综上所述，对于公交排班调度的两个流程，即公交时刻表排班与公交车辆调度，通过结合交通大数据进行方案设计，优化排班调度方案，可以提升公交系统的运行效率，降低公交公司的运营成本，提升乘客的出行体验，具有很强的现实意义。

1.2 国内外研究现状和相关工作

1.2.1 传统公交时刻表排班方法

启发式算法在公交时刻表排班中有着广泛的应用，Ceder 等学者[6] 使用启发式算法生成公交时刻表，以最大限度地实现公交车之间的同步运行。Han 等学者[7] 采用遗传算法进行多目标优化，在同时考虑公交公司的运营成本和乘客体验的情况下生成公交时刻表。Gkiotsalitis 等学者[8] 在考虑乘客出行时间的不确定性的基础上，设计了一种新的用于公交时刻表优化的遗传算法。Tang 等学者[9] 使用非支配排序遗传算法-II 来快速计算公交时刻表优化模型的帕累托最优解。Yu 等学者[10] 利用遗传算法优化公交发车间隔，缩短了乘客的平均等待时间。

此外数学规划法也被应用在公交时刻表排班问题中。Ceder[11] 使用公交运行的路线数据来减少所需要的公交班次和时刻表上的公交数量。Wihartiko 等学者[12] 使用一种基于改进遗传算法的整数规划模型求解公交时刻表排班问题。Shang 等学者[13] 提出了一种优先考虑乘客满意度的公交时刻表排班方法，用于优化公交车发车频率。

以上提到的方法均只能生成固定的公交时刻表，无法根据实时的客流变化动态进行调整。

1.2.2 基于强化学习的公交时刻表排班方法

近年来，也有一些研究使用强化学习解决公交时刻表排班问题。Yan 等学者[14] 提出了一种多智能体强化学习网络，在天气等不确定因素的影响下对公交时刻表进行优化，从而降低公交系统的运营成本。Li 等学者[15] 提出了一种基于已有的人

类经验的深度强化学习方法，动态优化下一次发车的时间间隔，以应对交通状况和乘客需求的突变。Zhao 等学者[16] 提出了一种深度强化学习方法，用于动态优化大人流量的公交线路的发车间隔，实现了以任意时间粒度进行公交调度。Ai 等学者[17] 提出了一种在每一分钟进行是否发车决策的深度强化学习方法，实现了能够应对突发客流变化的在线公交时刻表调度。

综上所述，强化学习在公交时刻表调度中有许多应用，但是以上所有方法都不能同时对一条公交线路的两个方向进行调度，也不能保证上下行发车次数相等。而在后续的公交排班调度中，有许多方案需要基于一个上下行发车次数相等的时刻表[18;19]。因此，本文中设计的可以同时进行双向公交时刻表排班的强化学习方法是十分有意义的。

1.2.3 静态环境下的公交车辆调度方法

在以往的研究中，大部分对于公交车辆调度的研究都无法应对可能出现的突发情况，这类静态环境下的方法可以分为使用精确算法和启发式算法两类。

使用精确算法的研究有许多，Chen 等学者[20] 使用混合整数规划，对校车的排班问题进行求解，用于最小化所需车辆数量和车辆的总行驶距离。Boyer 等学者[21] 提出了一个混合整数线性规划模型，在考虑车辆调度和驾驶员调度的情况下灵活进行排班。Janovec 等学者[22] 提出了一个线性数学模型，在考虑电动公交车需要充电的条件下对电动公交车进行调度安排。Gkiotsalitis 等学者[23] 提出了一种用于电动公交车的带时间窗的多车辆调度问题的混合整数非线性规划模型。

此外也有许多使用启发式算法的研究，Wen 等学者[24] 提出了一种混合整数规划公式和自适应大邻域搜索启发式算法，在对电动公交车进行排班的时候最少化所需车辆数量与总行程。Yao 等学者[25] 提出了一种启发式算法在考虑电动公交车的充电时间和能耗差异的情况下进行车辆调度。Yang 等学者[26] 以乘客总等待时间最少与公交利用率最大为优化目标，使用非支配排序遗传算法-II 来计算公交车辆调度问题的帕累托解集。

以上方法都是基于静态环境下的公交车辆调度算法，若出现了交通堵塞等突发问题，无法进行动态的调整。

1.2.4 不确定环境下的公交车辆调度方法

为了应对公交运营中出现的突发问题，也有一些研究提出了在不确定的环境下进行公交车辆调度的方法。Shen 等学者[27] 提出了一种鲁棒公交车辆调度方法，

使用一种具有随机公交行程时间的网络流模型增强了调度的鲁棒性。Tang 等学者[28] 提出了一种带有缓冲策略的静态模型来消除公交行程随机性造成的影响，同时提出了一种通过在固定时间中重新进行排班的动态方法来应对一天中不同时间的环境差异。Wang 等学者[29] 使用非支配排序遗传算法-II 对出现交通拥堵问题的时间段进行重新调度。

以上提到的鲁棒算法为了保证其鲁棒性，可能降低车辆的利用率从而导致运营成本的增加，而重调度算法需要频繁更新公交车辆调度方案，并且无法保证排班方案的性能。Liu 等学者[18] 首次将公交车辆调度问题建模为一个马尔可夫决策过程，通过深度强化学习对该问题进行求解，实现了在不确定环境下高效的公交车辆调度。本文基于该方法进行了一定的修改，实现了在出现交通拥堵情况下可实时进行调整的公交车辆调度。对比静态环境下的公交车辆调度方法，本文提出的方法可以更好地应对突发情况。

1.3 本文的研究内容与主要工作

本文研究内容为基于交通大数据的公交排班调度问题。在公交系统的工作流程中，公交时刻表排班问题与公交车辆调度问题是前后两个环节，一个完整的公交排班调度方案需要同时考虑这两个环节。公交时刻表排班问题即对于一条公交线路，何时需要在首站发出一辆公交车，本文研究的公交时刻表排班问题为一条线路的两个方向上的同时排班。公交车辆调度问题即在已知一个公交时刻表的基础上，如何选择车辆完成该时刻表。

为了解决现有的公交时刻表排班方法中无法在客流动态变化的情况下进行在线排班，并实现上下行发车次数相等的问题，本文提出了基于深度强化学习的双向动态公交时刻表排班算法（Deep Reinforcement Learning-based dynamic bus Timetable Scheduling method with Bidirectional Constraints，DRL-TSBC），将双向公交时刻表排班问题建模为一个马尔可夫决策过程，并提出了一个基于深度强化学习的解决方案。本文将公交车的利用率、乘客的等待时间和上下行发车次数作为状态，同时为了引导深度 Q 网络（Deep Q-Network，DQN）做出决策，本文设计了一个包含双向发车次数、公交利用率、乘客等待时间和滞留乘客数量的奖励函数。DRL-TSBC 在离线时刻表排班与在线时刻表排班中均有着优异的性能，同时可以保证上下行的发车次数相等。此外，当客流量出现了突变时，DRL-TSBC 可以根据变化的客流做出实时调整。本文使用真实世界中的数据对比了 DRL-TSBC、现在实际在公交线路上应用的人工方案和其他算法的性能。实验结果表明 DRL-TSBC

在双向公交时刻表排班中有着出色的性能表现。

为了在安排公交车辆完成时刻表时能够使用最少的车辆数量、尽量使得每一辆公交能够最终回到原出发站点，与此同时应对运营过程中出现的交通拥堵问题，本文在 Liu 等学者[18] 研究的基础上进行了一定的修改，提出了一种基于深度强化学习的公交车辆调度算法（Deep Reinforcement Learning-based Bus Scheduling Algorithm，DRL-BSA），其将公交车辆调度问题视为一个序贯决策问题，将公交时刻表上的每一个出发时刻点视为一个决策点，使用竞争深度双 Q 网络（Dueling Double Deep Q Network，D3QN）在每一个决策点选择车辆完成此发车计划，在 D3QN 网络中，加入了一个掩膜层用于避免选择当前时刻不可用的车辆[18]。本文将公交车辆调度问题建模为一个马尔可夫决策过程，设计了用于该解决问题的状态空间与奖励函数，其中奖励函数包含最终奖励与即时奖励。本文使用真实世界数据对 DRL-BSA 进行了测试，实验表明 DRL-BSA 可以在使用最少的车辆数量完成发车计划的同时，尽量使得所有车辆的发车次数均为偶数次。在出现道路拥堵时，DRL-BSA 可以在不增加车辆数量的情况下进行调整，以应对出现的突发情况。

1.4 本文的论文结构与章节安排

本文共分为四章，各章节内容安排如下：

第一章为引言部分，介绍了本文的选题背景与意义，国内外研究现状与相关工作，以及本文的研究内容与主要工作。

第二章介绍针对公交时刻表排班问题所提出的基于深度强化学习的双向动态公交时刻表排班算法（DRL-TSBC），首先介绍 DRL-TSBC 的研究问题建模方法、马尔可夫决策过程建模方法和具体实现的算法。随后介绍 DRL-TSBC 的离线排班测试结果和与其他算法的性能对比，同时展示 DRL-TSBC 对于动态客流下的在线排班性能，并给出 DRL-TSBC 的消融实验结果。

第三章介绍针对公交车辆调度问题所提出的基于深度强化学习的公交车辆调度算法（DRL-BSA），首先介绍 DRL-BSA 的研究问题建模方法、马尔可夫决策过程建模方法和具体实现的算法。随后介绍 DRL-BSA 的测试结果，并展示发生交通拥堵时 DRL-BSA 对车辆的实时调整能力，同时给出 DRL-BSA 的消融实验结果。

在第四章中，对本文所完成的工作进行总结，并对未来公交规划问题中使用的强化学习算法的研究方向进行展望。

第 2 章 基于深度强化学习的双向动态公交时刻表排班

2.1 公交时刻表排班问题

双向的公交时刻表排班问题包含两个部分：如何实现一条公交线路的单个方向上的动态公交时刻表排班，与如何保证一条公交线路的两个方向的发车次数一致。在公交系统中，大部分的公交线路都包含两个方向：上行与下行。对于这两个方向来说，公交车站的总数可能不一样，或出现上下行站点不对称的情况，但在一条公交线路上，首末站一般来说是一样的。在生成公交时刻表时，需要对当前客流情况做出判断，决定是否进行发车。同时，公交公司会对最大发车间隔 T_{max} 和最小发车间隔 T_{min} 进行限制，只有当本次发车与最近一次发车的间隔在一定范围内才允许发车。对于一条有两个方向的公交线路，保证上下行发车次数相等是必要的。只有保证上下行发车次数一致，后续在进行公交车辆安排时，所有的公交车辆才能回到原出发站点，这样可以避免车辆需要空载返回。在现实的公交系统中，存在一些特殊线路上行方向的最终站和下行方向的首发站是不同的，这些特殊的线路往往需要使用特殊的时刻表排班方法，因此本文只讨论首末站相同的公交线路。

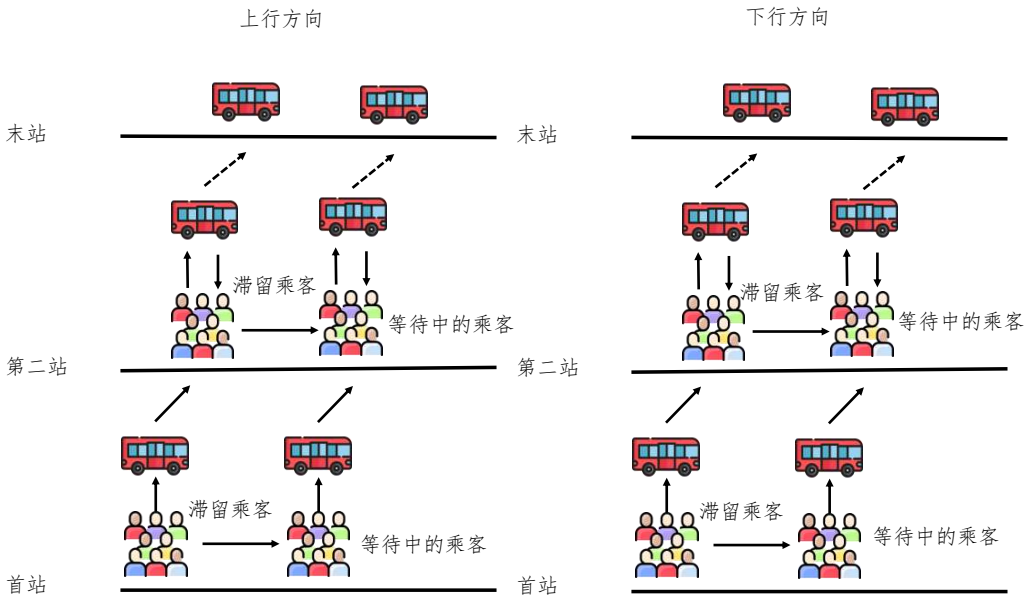

图 2-1 公交线路中的公交车移动与客流运动方向

生成公交时刻表时，对于上下行两个方向都需要根据当前时刻的客流数据来决定是否进行发车。如果确定某一方向需要发车，则从该方向的首发站向该方向发出一辆车。如图2-1所示，随着时间的推移，这辆公交车将在线路上向前移动，在

到达一个站点后，需要下车的乘客会先下车，随后需要上车的乘客会登上这辆公交。在这个时候，可能有一些乘客会由于公交车已经满载了而无法上车，本文将这些乘客称为被滞留的乘客，他们需要等待下一辆公交才能离开。在公交运营中，如果出现了滞留乘客，将会大大降低乘客的满意程度[30]，因此需要尽可能地减少滞留乘客的数量。

在搭建的仿真环境中，公交系统会以分钟为粒度进行模拟，该粒度与其他的一些已有算法是一致的[9;17]，可以较好地模拟出现实中的公交与客流变化情况。在这个环境中，包含了当前公交线两个方向的客流信息，一个控制器将在每分钟决定是否需要进行发车。同时，仿真环境将模拟乘客的上下车情况，并根据交通状况数据模拟线路上的公交运行情况。控制器将根据当前这一分钟的环境信息做出决策，并将当前决策应用于该环境中，以控制仿真系统是否进行发车。

在这个问题中，使用的客流数据包括乘客的上车站点、上车时间、下车站点以及估计的到达上车站点的时间。在描述交通状况时，我们将第 m 分钟，从第 k 个站点到达第 $k+1$ 个站点所需的时间记为 t_m^k，m 是每天的第 m 分钟，例如第 420 分钟为早上 7 时。

2.2 公交时刻表排班问题的马尔可夫决策过程建模方法

本文将公交时刻表排班问题建模为马尔可夫决策过程，其中的状态空间、动作空间和奖励函数的设计如下。

2.2.1 状态空间

智能体需要从状态空间中获取当前环境的重要特征，才能进行高效的学习。为了表示公交线路两个方向上的不同状态，将第 m 分钟的状态定义如下：

$$s_m = [a_m^1, a_m^2, x_m^1, x_m^2, x_m^3, x_m^4, y_m^1, y_m^2, y_m^3, y_m^4] \tag{2.1}$$

在状态空间中，所有的状态都被归一化在 $[0, 1]$ 区间内。由于在不同时刻可能出现相似的状态，所以在状态空间中需要加入表示时间的状态 a_m^1、a_m^2，定义如式 (2.2) 和式 (2.3)所示，

$$a_m^1 = t_h/24, \tag{2.2}$$

$$a_m^2 = t_m/60, \tag{2.3}$$

其中 t_h 是当前时间的小时数，t_m 是当前时间的分钟数。例如上午 7 时 30 分，$t_h = 7$，

$t_m = 30$。

为了表示上行与下行两个方向的信息，本文设计了一对结构相同的状态，分别表示为 x_m 和 y_m。其中 x_m 表示上行方向的状态，y_m 表示下行方向的状态。这两个组状态在构造上是一样的，下面将以状态 x 为例介绍这些状态的结构。

在公交系统中，公交车的满载率是一个非常重要的指标，其反映了乘客的需求量大小，满载率可由式 (2.4) 得出，

$$x_m^1 = C_{max}^{m,up}/C_{max}, \tag{2.4}$$

其中 $C_{max}^{m,up}$ 是第 m 分钟发出的公交所能达到的最大断面客流，即该公交在行驶过程中车上人数最多时候的乘客数量。C_{max} 是公交的最大载客量。当 $x_m^1 = 1$ 时，意味着此公交出现了满载的情况，并且可能出现了乘客因车辆满载而被滞留。

x_m^2 为归一化的乘客等待时间，由式(2.5)计算，

$$x_m^2 = W_m^{up}/\mu, \tag{2.5}$$

其中 W_m^{up} 是所有乘坐第 m 分钟发出的公交的乘客的等待时间之和，这个总等待时间可由式(2.6)计算得到，

$$W_m = \sum_{k=1}^{(K-1)} \sum_{i=1}^{l_m^k} (t_b^{m,i,k} - t_a^{m,i,k}), \tag{2.6}$$

其中 $t_a^{m,i,k}$ 和 $t_b^{m,i,k}$ 表示乘客 i 到达第 k 个车站的时间和坐上第 m 分钟发车的公交的时间。K 是这个方向上公交站的总数。l_m^k 是第 m 分钟在第 k 个车站上车的乘客总数。μ 是将 x_m^2 归一化至 $[0, 1]$ 区间内的一个参数。

公交车的客运容量是一个重要的参数，它反映了当前时刻进行发车是否是必要的，因此 x_m^3 也需要作为一个重要的状态。

$$x_m^3 = o_m^{up}/e_m^{up} \tag{2.7}$$

o_m/e_m 是客运容量的利用率[17]，对于一辆从第 m 分钟发车的公交，其所能提供的容量由式(2.8)得出，

$$e_m = \alpha \times C \times (K-1), \tag{2.8}$$

其中 K 是当前方向的车站数量，C 是公交车上的座位数。由于公交上除了座位，

还有中间的空间可以站立，所以实际上一辆公交可以承载的人数为 $\alpha \times C$。此处，α 为 1.5[31]。o_m 是第 m 分钟发出的公交实际被消耗的客运容量，即这辆公交在每一个站点时车上的乘客数量之和，由式(2.9)计算得到，

$$o_m = \sum_{k=1}^{K-1}(c_m^k + l_m^k - h_m^k), \tag{2.9}$$

其中假设有一辆公交在第 m 分钟从首发站发出，当它到达第 k 个车站时，c_m^k 表示此时车上的乘客数量，l_m^k 表示上车的乘客数量，h_m^k 表示下车的乘客数量。

此外，为了引导智能体在决策时实现两个方向上的发车次数相等，本文在状态函数中给出了相关的指引，所以当前时刻的两个方向的发车次数的差异也需要作为状态，其由式(2.10)计算，

$$x_m^4 = c_m^{up}/\delta, \tag{2.10}$$

其中 c_m^{up} 是在第 m 分钟时上行方向已经发出的公交数量，δ 是将 x_m^4 归一化至 $[0,1]$ 区间内的一个参数。

类似的，可以定义下行方向的状态，其由式(2.11)至式(2.14)计算得到。

$$y_m^1 = C_{max}^{m,down}/C_{max} \tag{2.11}$$

$$y_m^2 = W_m^{down}/\mu \tag{2.12}$$

$$y_m^3 = o_m^{down}/e_m^{down} \tag{2.13}$$

$$y_m^4 = c_m^{down}/\delta \tag{2.14}$$

2.2.2 动作空间

为了描述智能体的决策情况，动作 a 可由向量 $a = (a^{up}, a^{down})$ 表示，其中，a^{up} 是上行方向的决策动作，a^{down} 是下行方向的决策动作。对于两个方向来说，可用 0 表示不进行发车，1 表示进行一次发车，例如 $a = (0,1)$ 表示只有下行方向发车，$a = (1,1)$ 表示两个方向都进行发车。

2.2.3 奖励函数

为了引导智能体实现目标，即对一条公交线路的两个方向进行动态时刻表排班，并保证两个方向的发车次数一致，本文设计了一个与最终目标强相关的奖励函数。在这个奖励函数中，当前时刻的客流对是否发车的决策影响可用公交客运容量利用率[31] 和滞留旅客数量[17] 来表示。同时，两个方向的发车次数是否一致可用两个方向发车次数的差值表示。

在双向公交时刻表排班问题中，需要考虑公交线路的两个方向，所以本文将奖励函数分为两个部分，r^{up} 和 r^{down}。在第 m 分钟时，总的奖励通过式(2.15)计算得到。

$$r_m = r_m^{up} + r_m^{down} \tag{2.15}$$

本文分别对两个方向的奖励函数进行计算，对于其中的一个方向来说，只有两个动作，即“发车”和“不发车”。在当前方向上，假设一辆公交车在第 m 分钟发车，这辆车能提供的客运容量为 e_m，实际消耗的客运容量为 o_m，则可定义公交客运容量的利用率 o_m/e_m。o_m/e_m 为一个小于 1 的数，当这个数较大时，则说明此时人流量较大，因此更需要在第 m 分钟进行一次发车。相反地，若 $1 - o_m/e_m$ 较大，则说明不进行发车更符合当前低人流量的状态。因此，如果当前时刻做出的动作为“发车”，则 o_m/e_m 将作为奖励函数的一部分，而如果做出的动作为“不发车”，则 $1 - o_m/e_m$ 将作为奖励函数的一部分。

如果当前时刻的动作为“不发车”，则在奖励函数中应考虑由此导致的乘客等待时间的增加。因此，如果动作为“不发车”，则在奖励函数中加入惩罚 $\omega \times W_m$。W_m 表示乘坐第 m 分钟发车的公交的乘客的等待时间之和。ω 是使 $\omega \times W_m$ 保持在 $[0, 1]$ 区间内的一个参数。

此外，如果出现了滞留乘客，将会大大降低乘客的满意程度[30]。因此，无论是否进行发车，都引入一个惩罚 $\beta \times d_m$，表示目前的时刻表已经导致了出现滞留乘客而带来的惩罚。d_m 表示由于第 m 分钟发出的公交满载而无法上车的乘客数量。当滞留乘客数量过多时，奖励将会变为一个负数[17]。

为了实现两个方向的发车次数相等，需要在奖励函数中评估两个方向发车次数的差异带来的影响。假设当前方向的发车次数多于另一个方向，如果该方向的动作是“不发车”，则将获得一个正的奖励，如果动作是“发车”，则会获得一个负的奖励。相反，如果当前方向的发车次数少于另一个方向，则“不发车”将带来一个负的奖励值，若“发车”则会获得正的奖励。这个奖励值通过 $\zeta \times (c_m^{up} - c_m^{down})$ 计算得到，其中 c_m^{up} 为第 m 分钟时上行方向的发车次数，c_m^{down} 为第 m 分钟时下行

方向的发车次数。

综上所述，我们用 r^{up} 表示上行方向获得的奖励值，r^{down} 表示下行方向获得的奖励值，若以 0 表示不发车，1 表示发车，则上行与下行的奖励函数可由式(2.16)和式(2.17)计算得到。

$$r_m^{up}=\begin{cases}1-(o_m^{up}/e_m^{up})-(\omega\times W_m^{up})-(\beta\times d_m^{up})+\zeta(c_m^{up}-c_m^{down}), & a^{up}=0\\(o_m^{up}/e_m^{up})-(\beta\times d_m^{up})-\zeta(c_m^{up}-c_m^{down}), & a^{up}=1\end{cases} \tag{2.16}$$

$$r_m^{down}=\begin{cases}1-(o_m^{down}/e_m^{down})-(\omega\times W_m^{down})-(\beta\times d_m^{down})\\-\zeta(c_m^{up}-c_m^{down}), & a^{down}=0\\(o_m^{down}/e_m^{down})-(\beta\times d_m^{down})+\zeta(c_m^{up}-c_m^{down}), & a^{down}=1\end{cases} \tag{2.17}$$

2.3 深度强化学习算法

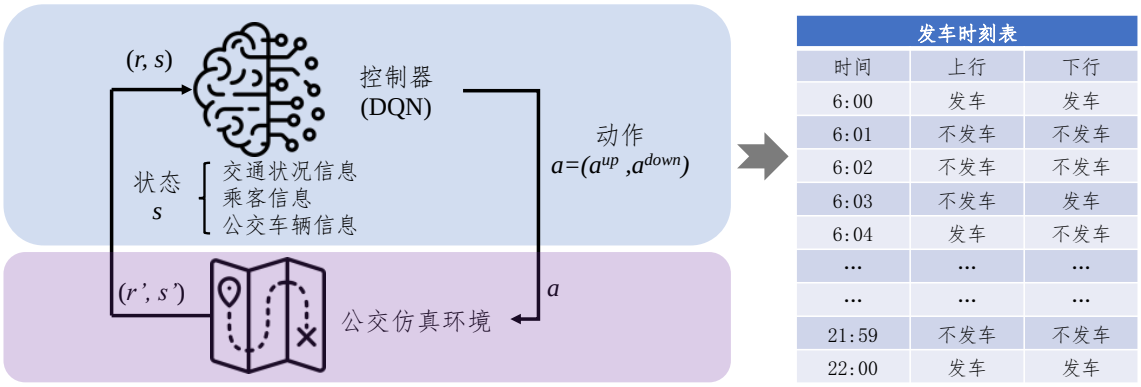

图 2-2 DRL-TSBC 对双向是否发车的控制流程

图2-2展示了 DRL-TSBC 对两个方向是否进行发车的控制。对于一条公交线路来说，其首班车和末班车的时间是固定的，在这两个时间固定会进行一次发车。在 DRL-TSBC 中，除了首末班车时间之外，控制器将根据这一分钟的状态选择一个动作，如果这个动作是进行发车，则对应的时间将会被记录在发车时刻表中。当控制器选择完一个动作以后，智能体将从仿真环境中获得下一分钟的状态和执行当前动作的奖励，同时以一定的频率进行学习。

由于需要同时处理公交线路的两个方向，双向动态公交时刻表排班问题有较大的状态空间，这可能导致维度灾难问题。为解决该问题，本文使用 DQN[32] 作为智能体，其结合了深度学习网络和 Q 学习，以解决离散动作空间下的状态空间动作决策问题。

与传统的强化学习方法相比，在 DQN 的学习过程中在解决高维状态空间和动作空间问题时具有更好的性能表现。同时，DQN 利用目标网络和经验回放，可以更好地平衡估计的误差和方差，提高算法的收敛性能。在 DQN 的学习过程中，其从仿真环境中积累经验，经验包括当前状态 s，动作 a，奖励 r，下一个状态 s'。并将这些经验以四元组 (s, a, r, s') 的形式储存在经验池 D 中。

算法 2.1: DRL-TSBC 的学习过程

超参数：经验池大小 M，批次大小 B，折现系数 γ，随机动作概率 ϵ，最大模拟次数 E，学习频率 P，模型更新频率 O，首班车时间 t_s，末班车时间 t_e
初始化经验池 D，主网络参数 θ，和目标网络参数 $\theta^- = \theta$
使用 θ 初始化主网络 $Q(s, a; \theta)$，并且使用 θ^- 初始化目标网络 $Q(s, a; \theta^-)$
对于所有 *episode* = *1* **转到** E **进行**
    初始化公交仿真环境
    对于所有 $i = t_s$ **转到** t_e **进行**
        如果 在 $[0, 1]$ 区间内随机取出的一个实数小于 ϵ **则**
            随机选择动作 a
        否则
            $a = \arg\max_a Q(s, a; \theta)$
        通过 a 计算 I^{up} and I^{down}
        如果 $I^{up} < T_{min}$ **则**
            $a_{up} \leftarrow 0$
        如果 $I^{up} > T_{max}$ **则**
            $a_{up} \leftarrow 1$
        如果 $I^{down} < T_{min}$ **则**
            $a_{down} \leftarrow 0$
        如果 $I^{down} > T_{max}$ **则**
            $a_{down} \leftarrow 1$
        $a \leftarrow (a_{up}, a_{down})$
        在环境中应用动作 a 并得到奖励 r 和下一个状态 s'
        将四元组 (s, a, r, s') 添加到经验池 D 中，同时如果 $|D| > M$，删去经验池中最旧的经验
        $s = s', i = i + 1$
        如果 $|D| > M$ 且 $i \mod P = 0$ **则**
            从经验池 D 中随机取出 B 个四元组 (s, a, r, s')
            通过式(2.18)计算损失函数 L
            使用 Adam 反向传播通过 L 更新 θ
            每进行 O 次学习，更新目标网络参数 $\theta^- = \theta$

DQN 具有两个结构相同的网络，即主网络与目标网络，两者均为全连接神经网络。在训练的过程时，从 D 中随机抽取一批已有的经验。目标网络使用固定的旧参数 θ^- 进行计算，参数 θ^- 每隔固定的训练轮次更新一次。在主网络和目标网

络之间优化均方误差，损失函数可由式(2.18)计算，

$$\mathcal{L}_i\left(\theta_i\right)=\mathbb{E}_{s,a,r,s'\sim D_i}\left[\left(r+\gamma\max_{a'}Q\left(s',a';\theta_i^-\right)-Q\left(s,a;\theta_i\right)\right)^2\right], \tag{2.18}$$

其中损失函数 $\mathcal{L}_i(\theta_i)$ 代表主网络和目标网络之间的误差。$Q(s',a';\theta_i^-)$ 是目标网络参数 θ_i^- 对下一个状态-动作对的预估 Q 值。$Q(s,a;\theta_i)$ 是主网络参数 θ_i 对当前的状态-动作对的预估 Q 值。γ 是未来奖励的折现系数。

对于一个方向来说，如果当前时间距离上次发车的间隔 I_m^{up} 或 I_m^{down} 小于最小发车间隔 T_{min}，则强制不进行发车；若其大于最大发车间隔 T_{max}，则强制进行一次发车。而在其他时刻，是否进行发车则由 DQN 进行决策。

在选择动作时，如果模型选择动作和公交公司给出的发车间隔约束相冲突，则对当前动作进行一次更新以使其满足约束。例如，如果 $a=(1,0)$，$I_m^{up}<T_{min}$，$I_m^{down}>T_{max}$，此时动作会被更新为 $a=(0,1)$，即只有下行方向进行一次发车。此外，本文使用 ϵ-贪婪策略来促进 DQN 进行探索。DRL-TSBC 的智能体学习过程的伪代码如算法2.1所示。

算法 2.2: DRL-TSBC 的推理过程

初始化公交仿真环境
对于所有 $i=t_s$ **转到** t_e **进行**
    $a=\arg\max_a Q(s,a;\theta)$
    通过 a 计算 I^{up} and I^{down}
    如果 $I^{up}<T_{min}$ **则**
        $a_{up}\leftarrow 0$
    如果 $I^{up}>T_{max}$ **则**
        $a_{up}\leftarrow 1$
    如果 $I^{down}<T_{min}$ **则**
        $a_{down}\leftarrow 0$
    如果 $I^{down}>T_{max}$ **则**
        $a_{down}\leftarrow 1$
    $a\leftarrow(a_{up},a_{down})$
    在环境中应用动作 a 并得到下一个状态 s'
    $s=s',i=i+1$
选择发车次数更多的方向
从时刻表中删除该方向的倒数第二次发车
$k\leftarrow$ 此时该方向的总发车次数
当 时刻表中的第 k 次发车和第 $k-1$ 次发车的间隔大于 T_{max} **进行**
    将第 $k-1$ 次发车的时间推迟直到其与下一次发车的间隔为 T_{max}
    $k=k-1$

由于规定了在末班车时间必须进行一次发车，并且发车时刻表需要满足最大

发车间隔和最小发车间隔的约束，所以使用 DQN 直接得到的公交时刻表往往上下行会出现发车次数相差一次的情况。因此在推理过程中，对最后几班车的发车时间进行调整，以实现两个方向发车次数相等。调整方法为从时刻表中删除发车次数更多的那个方向的倒数第二个发车计划，随后以 T_{max} 为发车间隔，向前调整公交的发车时刻，直到不需要再进行调整，即满足发车间隔的约束。推理过程的伪代码如算法2.2所示。

2.4 数据集与实验设置

2.4.1 真实世界的数据集

本文使用的数据集为 2023 年 8 月 14 日 A 市 208、211 线的乘客刷卡数据，每一条数据包含乘客唯一识别码、上车站点、下车站点和上车时间。

为了计算乘客等待公交的时间，需要估算出每一个乘客到达上车站点的时间，本文使用正态分布对乘客到站时间，结合当天线路上各个公交站的车辆进站和出站时间数据进行推测。正态分布的均值为乘客的上车时间，标准差为上车时间与上一辆离开该站点的公交的离站时间之差的二分之一。此外，这些线路的公交的座位数设为 32，α 设为 1.5。数据集的具体信息如表2-1所示。

表 2-1 公交线路信息

线路	方向	营运时间	站点数量	乘客数据集数量
208	上行	6:00-21:00	26	3157
208	下行	6:00-21:00	24	2604
211	上行	6:00-22:00	17	2266
211	下行	6:00-22:00	11	2126

2.4.2 进行对比的算法

DRL-TSBC 是一种在一条公交线路的两个方向上进行实时公交时刻表排班的算法。公交时刻表排班可分为离线排班和在线排班，离线排班指根据以往的数据进行一次排班，随后不对排班结果进行修改，而在线排班指根据实时的客流变化，动态修改公交时刻表。对于离线排班结果，本文采用 A 市公交目前正在使用的人工方案进行对比，该方案所得到的时刻表也具有上下行发车次数相等的特点。此外，本文还与 DRL-TO 算法[17] 进行了对比，DRL-TO 算法使用深度强化学习对单个方向进行公交时刻表排班，本文将会评估其是否能够实现上下行发车数量相等。

对于在线排班，目前还没有其他算法可以在两个方向上实现实时在线排班，并

保持上下行发车次数相等。因此，本文将只展示 DRL-TSBC 对实时客流变化的应对能力。

2.4.3 实验设置

本文搭建了一个可用于双向公交运行仿真的环境，通过乘客刷卡数据来模拟实时客流，并通过离线学习的方法训练 DRL-TSBC，状态空间与奖励函数中的参数为 $\mu = 5000, \delta = 200, \beta = 0.2, \zeta = 0.002$。训练结束后，DRL-TSBC 会被应用到在线时刻表排班中，即根据当前环境的状态做出实时决策。人工方案为目前实际应用在公交线路上的方案。DRL-TO 使用单向公交仿真环境进行训练和测试。

DQN 网络的结构为一个全连接网络[32]，网络的参数如表2-2所示。所有方法的测试均在笔记本电脑上完成，CPU 型号为 AMD Ryzen 9 7945HX，GPU 型号为 NVIDIA RTX 4060 Laptop，内存大小为 32GB。DRL-TSBC 和 DRL-TO 均使用 Python 中的 Pytorch 库实现。

表 2-2 DQN 网络的参数

参数	值
学习率	0.001
权重初始化方法	正态分布初始化
激活函数	ReLU
隐层数量	12
隐层神经元数量	500
批次大小 B	64
折现系数 γ	0.4
经验池大小 M	3000
ϵ	0.1
最大模拟次数 E	50
学习频率 P	5
参数更新频率 O	100

2.5 离线时刻表排班实验结果

表2-3展示了人工方案、DRL-TO 和 DRL-TSBC 生成的时刻表的性能。其中对比了乘客平均等待时间、发车次数和被滞留的乘客数量。乘客平均等待时间和发车次数是一对相互矛盾的指标，更多的细节将在第2.7节进行讨论。通过调整式(2.16)和式(2.17)中的 ω，可以得到发车次数接近于人工方案的结果。

由于 DRL-TO 无法实现上下行发车次数一致的时刻表排班，因此通过不断修改 ω，使其上下行发车次数尽可能保持相等，同时接近于人工方案的发车次数。而

表 2-3 人工方案、DRL-TO 和 DRL-TSBC 的时刻表排班实验结果

线路		208		211	
方向		上行	下行	上行	下行
人工方案	发车次数	72	72	76	76
	乘客平均等待时间 (m)	4.06	4.6	4.68	3.5
	被滞留乘客数量	3	176	0	0
DRL-TO	发车次数	69	72	75	76
	乘客平均等待时间 (m)	3.7	3.2	3.6	3.1
	被滞留乘客数量	0	7	0	0
	ω	1/1000	1/500	1/500	1/300
DRL-TSBC	发车次数	73	73	75	75
	乘客平均等待时间 (m)	3.7	3.8	4.0	3.3
	被滞留乘客数量	0	0	0	0
	ω	1/1000		1/900	

在实际公交时刻表排班中，通过不断调整 ω 使得上下行发车次数相等是不现实的，因为修改 ω 意味着需要重新进行训练，对于拥有大量公交线路的公交系统来说，这样做是十分消耗时间的，同时也不一定能够保证能得到上下行发车次数相等的结果。

从表2-3中可以看出，当发车次数接近的时候，DRL-TO 和 DRL-TSBC 都能更好的安排时刻表，实现更短的乘客平均等待时间，并且几乎不会出现乘客被滞留的现象。与 DRL-TO 相比，DRL-TSBC 生成的时刻表稍微增加了乘客的平均等待时间，这是由于在设计奖励函数时，DRL-TSBC 引入了保持上下行发车次数相等的奖励。但是和人工方案相对比，乘客平均等待时间仍有明显降低。略微增加乘客的平均等待时间以实现上下行两个方向的发车次数相等，是非常有意义的。

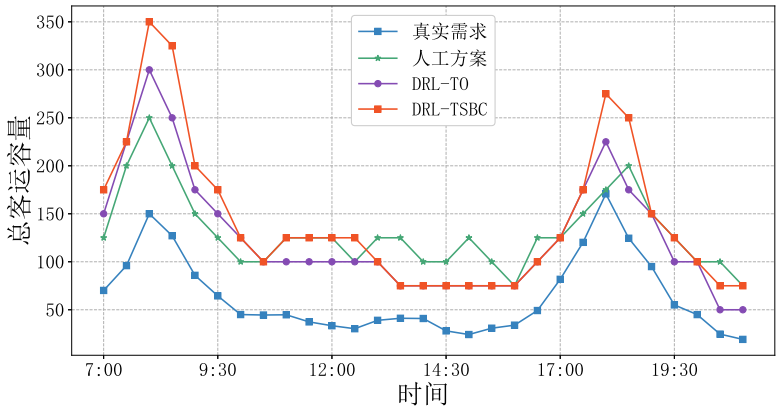

图 2-3 不同算法生成的公交时刻表所提供的总客运容量在 208 线上行方向的对比

图2-3将乘客的实际需求量与不同方法生成的公交时刻表所提供的总客运容量进行了对比。乘客的实际需求为当前时刻线路所有公交上的乘客数量，而总客运

容量为当前线路上的所有公交的荷载人数之和。可以看到，DRL-TSBC 和 DRL-TO 在高峰时段比人工方案提供了更大的客运容量，即发车更加频繁，而在非高峰时段则比人工方案提供更少的客运容量，即发车频率更低。这也解释为什么 DRL-TSBC 可以缩短乘客的平均等待时间。在所有的测试中，DRL-TSBC 与人工方案相比，平均乘客等待时间降低了 12.1%，并有效减少了滞留乘客数量。

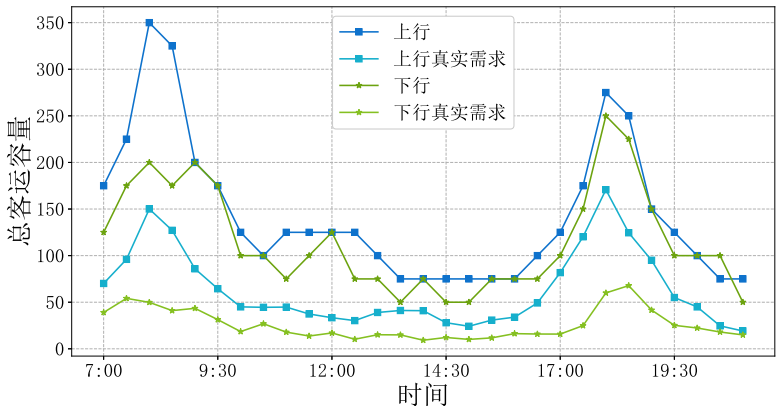

图 2-4 DRL-TSBC 在 208 线上下行方向生成的公交时刻表所提供的总客运容量与真实需求的对比

图2-4为 208 线上下行方向 DRL-TSBC 的排班结果和实际需求的对比，可以看出，DRL-TSBC 可以很好的满足两个方向的实际需求，并在两个方向的实际需求存在一定差异的情况下，实现两个方向的发车数量相等。

2.6 在线时刻表排班实验结果

DRL-TSBC 在线时刻表排班采用训练好的模型生成实时公交时刻表。DRL-TSBC 将根客流的变化做出调整，得到与客流情况相匹配的发车时刻表，并保证上下行发车数量相等。

为了模拟突发事件导致的客流突增，我们在 208 线的上行方向的第九个站点增加了晚高峰期间的客流量。我们使用 DRL-TSBC 进行实时推理，得到的结果如图2-5所示。可以看到在晚高峰时段，DLR-TSBC 在相应的时间段提供了更多的客运容量以应对增加的客流，同时在晚高峰时段增加上行的客流量并保持下行客流量不变的情况下，DRL-TSBC 得到的双向发车次数为 76 次，仍然保持相等。

如果对晚高峰提前的情况进行模拟，DRL-TSBC 得到的调度结果如图2-6所示。可以看到随着晚高峰时间的提前，DRL-TSBC 更早地提供了更多的发车次数来应对这一变化。图2-7为提前了上行方向晚高峰时间后，上下行方向的调度结果和真实需求的对比。可以看出，DRL-TSBC 对下行方向的调度不受上行变化的影响，下

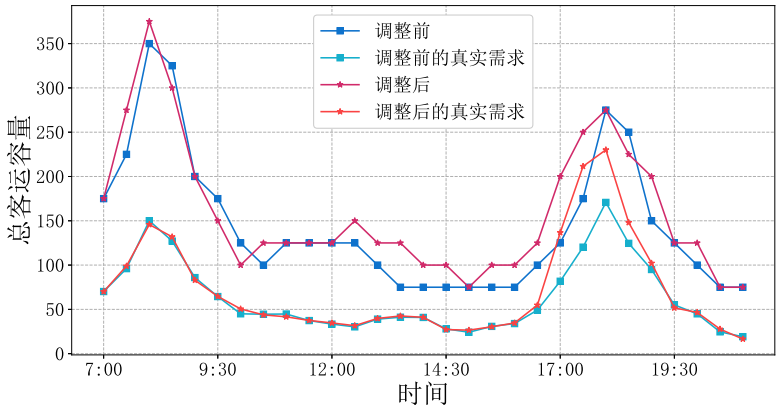

图 2-5 增加晚高峰客流前后 DRL-TSBC 生成的公交时刻表提供的总客运量与真实需求的对比

行方向的晚高峰时间仍然保持在原来的时间段。同时，DRL-TSBC 得到的两个方向的发车次数均为 75 次，仍保持相等。

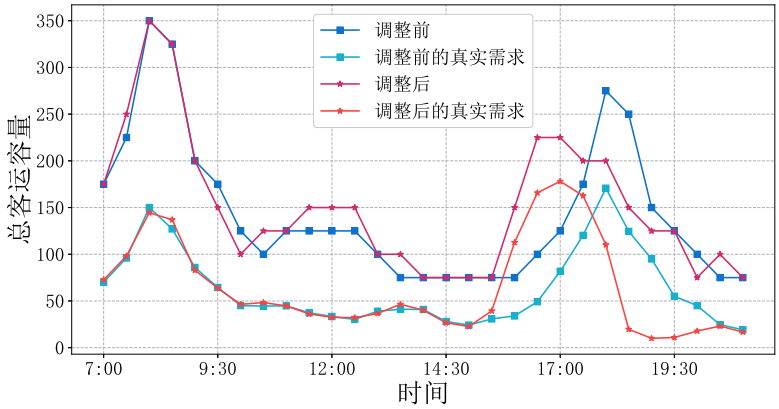

图 2-6 提前晚高峰前后 DRL-TSBC 生成的公交时刻表提供的总客运量与真实需求的对比

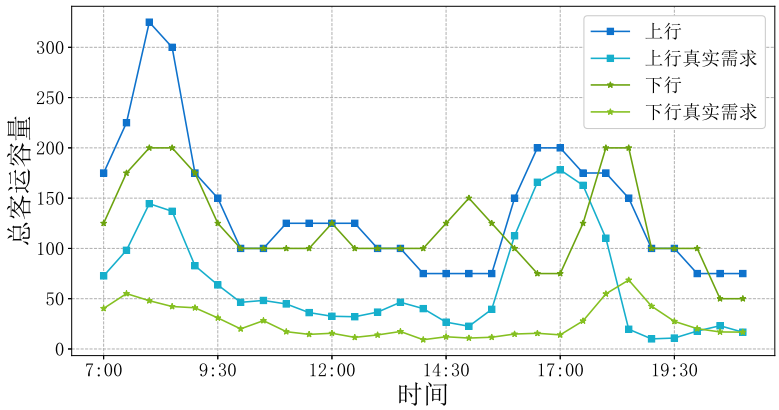

图 2-7 提前晚高峰前后 DRL-TSBC 生成的公交时刻表提供的上下行客运容量对比

在线调度实验表明，DRL-TSBC 可以在客流量实时变化的情况下动态满足乘

客的需求，同时实现上下行发车次数一致。

2.7 不同 ω 下的实验结果

在本节中，将会讨论 DRL-TSBC 在不同 ω 值下的性能，并评估其是否能稳定实现上下行发车次数相等。使用不同的 ω 对 208 线和 211 线进行了测试，得到的结果如图2-7和表2-4所示。可以看到，当 ω 不同时，DRL-TSBC 总能实现上下行发车次数相等。当 ω 减小时，两个方向的发车次数同时减少并保持一致，而由于乘客的平均等待时间（average waiting time，AWT）和发车次数是一对互相矛盾的特征，因此随着 ω 的减少，乘客的平均等待时间会增加。

这意味着在保证上下行发车次数相等的同时，公交公司可以通过调整 ω 来平衡乘客的出行体验与公交运营成本。

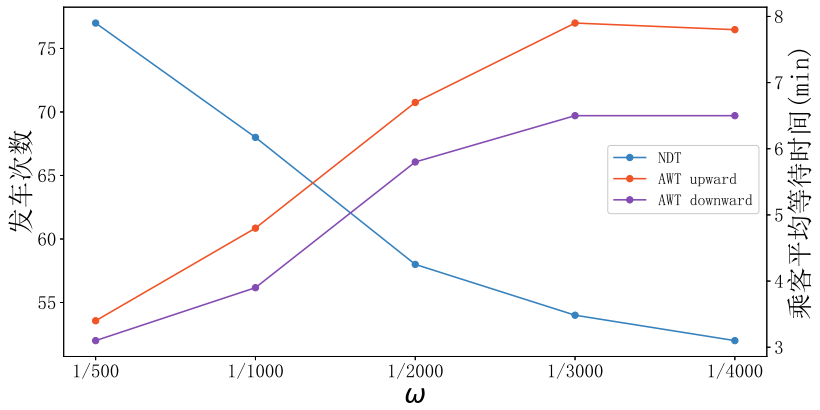

图 2-8 DRL-TSBC 在不同 ω 下的 211 线发车次数与乘客平均等待时间的对比

表 2-4 DRL-TSBC 在不同 ω 下的测试结果

	208			211		
ω	发车次数	上行 AWT	下行 AWT	发车次数	上行 AWT	下行 AWT
1/500	89	2.6	2.5	77	3.4	3.1
1/1000	77	3.5	3.3	68	4.8	3.9
1/2000	72	3.8	4.0	58	6.7	5.8
1/3000	70	4.7	4.2	54	7.9	6.5
1/4000	64	4.9	4.9	52	7.8	6.5

2.8 消融实验

在 DRL-TSBC 中，第2.2.1节中设计的状态空间与第2.2.3节中设计的奖励函数，都是为了实现上下行发车次数一致的实时公交时刻表排班。为证明其有效性，进

行如下的实验。

为了实现上下行发车次数相等，本文在奖励函数中设计了 $\zeta \times (c_m^{up} - c_m^{down})$ 来引导智能体实现一致的发车次数。为了验证其有效性，将 ζ 设为 0 后进行测试，结果如图2-9所示，可以看到在多次训练后，两个方向的发车次数仍然有较大差距。

为了让智能体能够感知到当前时间上下行发车次数的差异，在状态空间中加入了 x_m^4 和 y_m^4 来表示发车次数。为了评估其对性能的影响，将这两个状态从状态空间中删除，随后进行测试。如图2-10所示，在经过多次训练后，该方案仍无法实现收敛，这意味着在状态空间中加入 x_m^4 和 y_m^4 是非常有必要的。

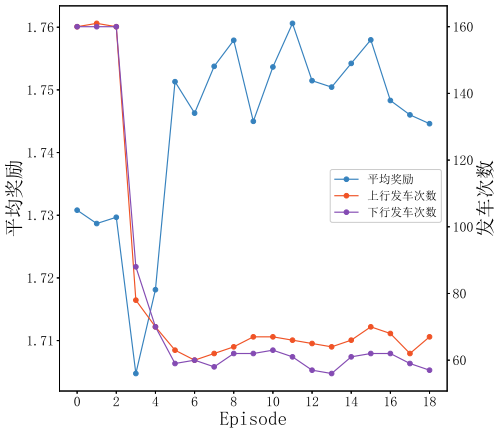

图 2-9 奖励函数中 $\zeta = 0$ 时的测试结果

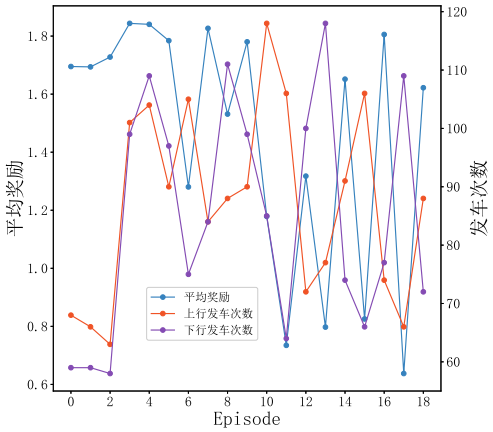

图 2-10 删去状态空间中 x_m^4 和 y_m^4 后测试结果

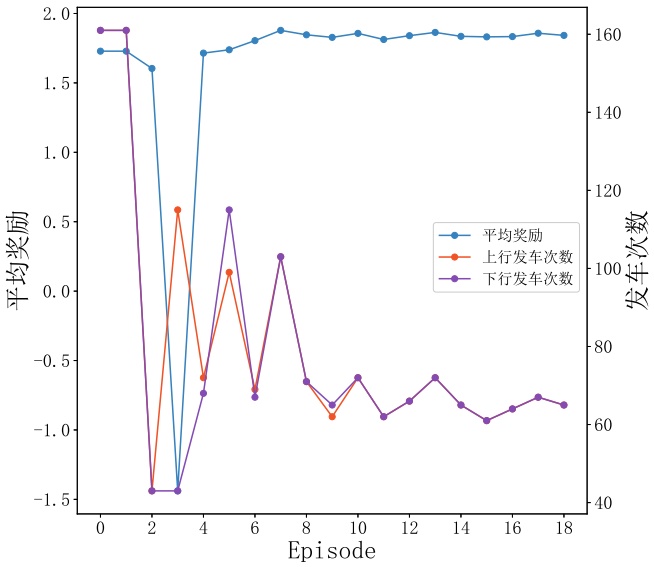

图 2-11 DRL-TSBC 的收敛结果

而在测试 DRL-TSBC 时，如图2-11所示，可以看到在经过一定的训练次数后，

DRL-TSBC 的平均奖励值趋于稳定，并且上下行的发车次数也保持相等，实验结果表明，DRL-TSBC 算法在双向公交时刻表排班中是十分有效的。

2.9 本章小结

在本章中，详细介绍了基于深度强化学习的双向动态公交时刻表排班算法，算法以分钟为粒度进行公交线路的模拟，对乘客和车辆的运动情况进行仿真。本文将公交时刻表排班问题建模为马尔可夫决策过程，其中的状态空间与奖励函数与双向动态公交时刻表排班是强相关的。本文使用 DQN 对每一分钟是否需要进行发车做出决策，并且引入了一个修正算法以确保最终上下行的发车次数是相等的。

对 DRL-TSBC 的离线排班与在线排班进行的测试结果表明，DRL-TSBC 对比现有的人工方案能够更好地降低乘客的平均等待时间，同时也能保持上下行发车次数的相等。并且当客流量发生变化时，DRL-TSBC 可以在不重新进行训练的情况下调整公交时刻表以满足乘客的出行需求。此外通过更改 ω，公交公司也可以很方便地调整策略，以平衡运营开销和乘客的满意程度。消融实验结果表明，状态空间和奖励函数的设计在双向动态公交时刻表排班中是十分有效的。

第 3 章 基于深度强化学习的公交车辆调度

3.1 公交车辆调度问题

公交车辆调度问题是指在一个已知的固定公交时刻表中，安排车辆完成时刻表上的每一个发车任务。公交时刻表上的每一个发车时间点都由唯一的一辆公交完成，时刻表上的每一个发车时间都是固定的，但是一辆公交车完成这个计划所消耗的时间取决于当前时刻的交通状况。公交车辆调度问题最终优化的目标是使得使用的车辆数量最少，并满足约束条件。

本文研究的公交车辆调度问题为首末站相同的公交线路上公交来回行驶的情况。对于一条公交线路，上下行各有一个首站，在两个首站中各有一个发车时刻表，记为 T_1 和 T_2，每个时刻表中均有若干个发车时间点，记作 $T_1 = \{t_1^1, t_1^2, t_1^3, \ldots, t_1^{N_1}\}$ 和 $T_2 = \{t_2^1, t_2^2, t_2^3, \ldots, t_2^{N_2}\}$，其中 N_1 和 N_2 分别为上下行时刻表的发车次数。一般来说，N_1 和 N_2 是相等的，这是保证在进行车辆调度时所有车辆都可以回到原出发点的前提条件。本文将两个方向的时刻表按照时间先后顺序进行合并，最终得到一张时刻表 T，T 中的每一个时间都是一个需要进行决策的时间点。

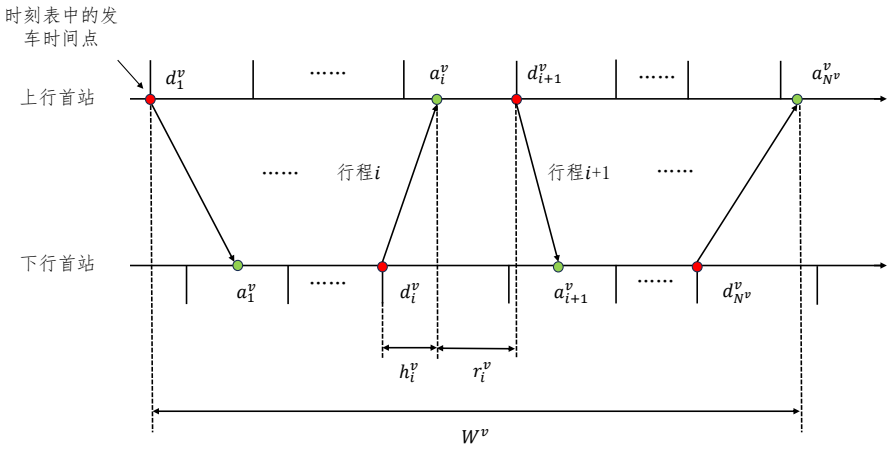

图 3-1 一辆公交车一天内的调度情况

对于一辆公交来说，其在线路上行驶的过程如图3-1所示，其每一个出发时间都是时刻表上的一个时间点，将车辆 v 第 i 次发车的出发时间记为 d_i^v，在经过一段时间后其会到达终点，到达的时间记为 a_i^v，对于这一次行程来说，车辆的驾驶时间为 $h_i^v = a_i^v - d_i^v$。而到达终点站的时间与这辆车下一次的发车时间之间的间隔

为 $r_i^v = d_{i+1}^v - a_i^v$，即休息时间。设车辆 v 一天中总共进行的行程数量为 N^v，则其在一天内的总驾驶时间为 $H^v = \sum_{i=1}^{N^v} h_i^v$，而总的休息时间则为 $R^v = \sum_{i=1}^{N^v} r_i^v$，由此，可以定义车辆 v 的总工作时间为 $W^v = H^v + R^v$。

对于公交车辆调度问题来说，需要满足几个约束。首先是在对于一辆公交车，在进行了一次行程后，其休息时间必须大于最短休息时间 r_{min}，即 $r_i^v \geq r_{min}$。其次是所有车辆的总驾驶时间不能大于一天的最大驾驶时间 H_{max}，即 $H^v \leq H_{max}$，且总工作时间不能大于一天的最大工作时间 W_{max}，即 $W^v \leq W_{max}$。而关于工作时间的约束是弱约束，在实际运营过程中可以允许车辆的工作时间略高于最大工作时间，本文中这个值被设为九十分钟。同时，一辆车辆的行程数量不可大于最大行程次数 N_t。公交车辆调度问题有两个优化的目标，最小化完成时刻表所需要的车辆数量 N_u 和最小化奇数次发车的车辆数量 N_o。在合并后的时刻表 T 中的每一个时间点，都需要从满足约束条件的可选择车辆中选择一辆车辆进行发车。

3.2 公交车辆调度问题的马尔可夫决策过程建模方法

本文将公交车辆调度问题建模为马尔可夫决策过程，其中的状态空间、动作空间和奖励函数的设计如下。

3.2.1 状态空间

在学习过程中，智能体会根据所有的可用车辆的状态进行决策，选择一辆车辆进行发车。智能体需要从状态空间中感知当前的可用车辆的信息以完成决策，在本文中，对于每一辆可用车辆 v，状态包含以下四个部分：

1) 已经进行的行程数量 n^v：表示车辆 v 已经完成了几次发车任务，可用于表示当前车辆是否回到了最开始的出发站点，如果 n^v 是一个偶数，则代表此车辆已经回到了最开始的出发站点。
2) 剩余工作时间 w_r^v：对于车辆 v，设其在当前时刻，累计的驾驶时间为 h^v，累计的休息时间为 r^v，则截至当前时刻，其工作时间为 $w^v = h^v + r^v$，则其剩余工作时间为 $w_r^v = W_{max} - w^v$。如果一辆车剩余工作时间很少，并且已经进行的行程数量不多，则需要尽量选择这辆车，否则会导致车辆利用率低的问题。
3) 剩余驾驶时间 h_r^v：除了需要满足工作时间约束外，车辆还需满足驾驶时间的约束，$h_r^v = H_{max} - h^v$，其与前两项一起反映了选择当前车辆的紧急程度。
4) 休息时间 r^v：为车辆 v 已经休息的时间，在公交车辆调度中，一般不希望出

现一辆车在等待了很久之后都没有进行下一次发车，这同样会导致车辆利用率低的问题。

车辆类型	h_r^v	r^v	n^v	w_i^v
V_a	110	5	8	50
	200	10	5	110
	…	…	…	…
V_n	0	0	0	0
V_u	−1	−1	−1	−1
	…	…	…	…

N_a

N

图 3-2 车辆信息矩阵

车辆信息的状态可用一个矩阵表示，如图3-2所示，在这个矩阵中，前 N_a 行包括了 N_a 辆可用车辆的状态信息，可用车辆使用集合 V_a 表示。集合 V_a 中的车辆均为在当前时间点前进行过至少一次行程的车辆。对于不同的决策时间点，可用车辆数量是不一样的，也就是说 N_a 的值是不确定的。而对于一个深度强化学习网络来说，要求输入的数据维度是固定的，因此需要对这个矩阵进行填充，使其中的状态数量保持恒定。首先在第 N_a+1 行，使用 0 进行填充，其表示一辆新的车辆，该车辆以前并没有完成过任何的行程，使用集合 V_n 表示。而往后的 $N-1-N_a$ 行，使用-1 进行填充，表示这些车辆是不可用的，使用集合 V_u 表示。处理后的车辆信息矩阵一共包含 N 行，在输入深度强化学习网络时，将其维度变为一维后就得到了一个具有 N 个元素的一维向量。

除了车辆信息以外，为了使得智能体能够感知到当前进行决策的车辆是从哪一个方向发出的，所以在上述一维向量的最后加入一个表示当前决策方向的状态，其取值为 0 或 1，0 表示当前决策方向为上行，即车辆要从上行方向发出，1 表示当前决策方向为下行。同时为了对不同时间的决策进行区分，在这个一维向量的最后再加入一个状态 m，表示当前决策的时间点，m 表示当前决策点的时间是这一天的第 m 分钟。

3.2.2 动作空间

在一个决策时间点，动作空间为 $V_a \cup V_n \cup V_u$，动作 a 即为选择一个车辆 v。若 $v \in V_a$，则代表选择了一辆以前曾经完成过至少一次行程的车辆。而如果 $v \in V_n$ 则代表选择了一辆新的车辆，这辆车辆以前没有完成过任何行程。如果 $v \in V_u$，则代表选择了一辆不可用的车辆，为了避免选择到不可用的车辆，在第3.3节中将会

提出一种掩膜机制，用于在选择动作时避免选择到不可用的车辆。

3.2.3 奖励函数

为了引导智能体实现最少化的车辆使用数量，奖励函数的设计至关重要，对于公交车辆调度问题，其排班结果的好坏取决于一整天结束以后得到的最终结果。而由于决策序列很长，如果只在最后给出一个奖励值，会导致稀疏奖励问题，使得算法难以收敛。因此本文设计了一个包含最终奖励和即时奖励的奖励函数，最终奖励会在最后一个决策点出现，而即时奖励则是在每一个决策点出现。

最终奖励奖励如下：

首先为了避免最终奖励是一个负数导致智能体不愿意进行探索，故首先在最终奖励中加入一项 N_r，使得最终奖励不恒为负数。为了使得使用的车辆数量最少，所以最终使用的车辆数 N_u 会作为最终奖励中的一个惩罚项。

由于公交司机一般居住在第一个出发站点的附近，所以让公交车在一天之后回到最开始的出发站点是非常重要的，为了使得车辆的行程数为偶数，将最终排班方案中行程次数为奇数的车辆数量 N_o 作为一个惩罚项。

为了提升车辆的利用率，将最终行程数量达到最大行程次数 N_t 的车辆数量 N_f 作为一个奖励项。同时为了减少不同车辆的总驾驶时间的差异，故将最终所有车辆的驾驶时间的标准差 σ 作为一个惩罚项。

综上，最终奖励 r_m 可通过式(3.1)计算得到，

$$r_m = w_1^1 \times N_r - w_2^1 \times N_u - w_3^1 \times N_o + w_4^1 \times N_f - w_5^1 \times \sigma, \tag{3.1}$$

其中 w_1^1、w_2^1、w_3^1、w_4^1、w_5^1 均为正实数。

为了使得智能体在进行每一次决策时能够获得一定的反馈，并引导其实现最终目标，即时奖励的设计如下：

首先为了避免即时奖励恒为负数，即时奖励中会有固定的奖励项 r_l。如果在选择车辆的时候选择了一辆新的车辆，这就意味着最终使用的车辆数量会增多一辆，所以当选择了新的车辆时，应该给予一个惩罚。若当前时刻选择新的车辆是因为 V_a 为空，即没有旧车辆可以选择，此时在即时奖励中会引入惩罚项 r_n，如果是在 V_a 不为空的情况下选择了新车辆，则引入惩罚项 r_e。

为了提高车辆利用率，智能体应该优先选择休息时间长的车辆。如果选择的车辆不是一辆新车，此时将 V_a 中的车辆的休息时间 r^v 进行降序排序，其中被选择的车辆的休息时间排在第 p^v 位。由于需要更倾向于选择休息时间更长的车辆，所

以在即时奖励中引入惩罚项 $r_k = p^v / N_a$。

如果在选择了一辆车辆后，在完成当前行程后，其剩余工作时间或剩余驾驶时间不足其完成下一次行程，且其行程次数为奇数次，则在即时奖励中加入惩罚项 r_o。

综上，即时奖励 r_b 可通过式(3.2)计算得到，

$$r_b = w_1^2 \times r_l - w_2^2 \times r_n - w_3^2 \times r_e - w_4^2 \times r_k - w_5^2 \times r_o, \tag{3.2}$$

其中 w_1^2、w_2^2、w_3^2、w_4^2、w_5^2 均为正实数。

3.3 深度强化学习算法

和公交时刻表排班问题一样，公交车辆调度问题的状态空间也非常大，传统的强化学习算法难以对其进行有效的学习。而使用经典的 DQN 算法可能存在 Q 值高估和训练不稳定的问题。在公交车辆调度问题中，本文使用竞争深度双 Q 网络（Dueling Double Deep Q Network，D3QN）[33] 作为深度强化学习网络。其结合了竞争深度 Q 网络（Dueling DQN）和深度双 Q 网络（Double DQN）[34]，可以有效解决 Q 值高估问题。

D3QN 网络包含两个结构一致的网络，即主网络和目标网络，在 D3QN 中，由于主网络和目标网络的结构是一样的，下面对主网络结构进行介绍。主网络的结构如图3-3所示。

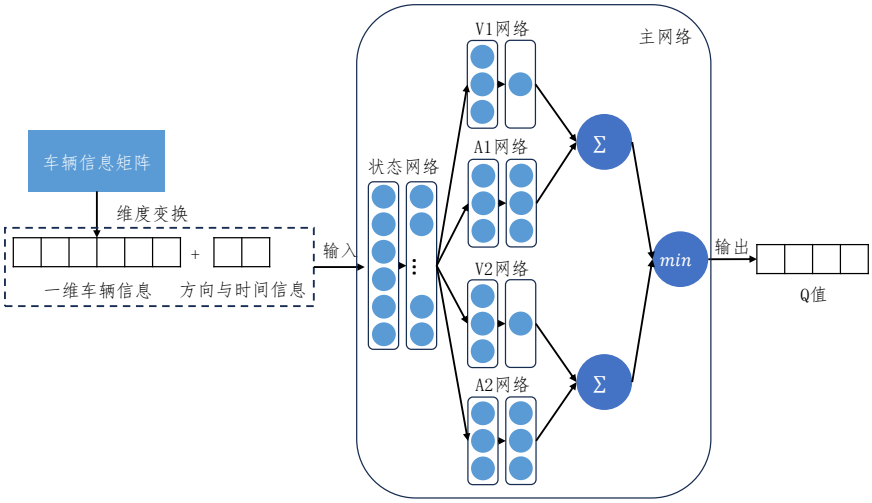

图 3-3 D3QN 网络结构

主网络包含三个部分，状态网络和两对状态网络（V 网络）和优势值网络（A 网络）。D3QN 将动作之分解状态值和优势值两个部分，一对状态网络和优势值网络所得到的 Q 值由式(3.3)计算得到，

$$Q(s,a,\theta)=V(s,\theta)+A(s,a,\theta) \tag{3.3}$$

V 是 V 网络的输出，代表了状态的价值，即预期的累积奖励，A 则是 A 网络的输出，代表了一个动作相比于其他动作的优势。对于每一个动作的 Q 值，都从这两对网络中选择一个更低的 Q 值进行输出，以减轻 Q 值的高估问题[35]。

图3-2中的车辆信息矩阵在经过平坦化后的维度数为 $4N$，方向、时间状态进行拼接后得到的输入向量维度为 $4N+2$，状态网络是一个两层的全连接神经网络，其与两对 V 网络和 A 网络相连，V 网络和 A 网络均是一个两层的全连接神经网络，其中 V1 网络和 V2 网络结构相同，A1 网络和 A2 网络结构相同。对于其中的一对 V 网络和 A 网络，其输出的 Q 值由式(3.3)计算得到，随后最终输出的 Q 值为两对网络中 Q 值的较小值。

为了避免在选择动作是选择到不可用的车辆，本文引入了一个掩膜机制。在选择动作时，将不可用的车辆的 Q 值设置为负无穷，这样在选择动作时就不会选择到不可用的车辆。对于输出的 Q 值来说，其末尾的若干个元素为代表 V_u 的 Q 值，即代表选择不可用的车辆，将这些元素设为负无穷即可避免智能体在选择动作时选择到不可用的车辆。

与 DQN 的类似，在 D3QN 的学习过程中，其从仿真环境中积累经验，并将这些经验以四元组 (s,a,r,s') 的形式储存在经验池 D 中。

在训练的过程时，从 D 中随机抽取一批已有的经验。目标网络使用固定的旧参数 θ^- 进行计算。在主网络和目标网络之间优化均方误差，损失函数可由式(3.4)计算，

$$\mathcal{L}_i\left(\theta_i\right)=\mathbb{E}_{s,a,r,s'\sim D_i}\left[\left(r+\gamma\max_{a'}Q\left(s',a';\theta_i^-\right)-Q\left(s,a;\theta_i\right)\right)^2\right], \tag{3.4}$$

其中损失函数 $\mathcal{L}_i(\theta_i)$ 代表主网络和目标网络之间的误差。$Q(s',a';\theta_i^-)$ 是目标网络参数 θ_i^- 对下一个状态-动作对的预估 Q 值。$Q(s,a;\theta_i)$ 是主网络参数 θ_i 对当前的状态-动作对的预估 Q 值。γ 是未来奖励的折现系数。

DRL-BSA 中 D3QN 的训练过程如算法3.1所示：

算法 3.1: DRL-BSA 的学习过程

超参数：经验池大小 M，批次大小 B，折现系数 γ，随机动作概率 ϵ，最大模拟次数 E，预训练步数 tp，衰减率 β，目标网络的更新率 α
初始化经验池 D，主网络参数 θ，和目标网络参数 $\theta^- = \theta$
使用 θ 初始化主网络 $Q(s,a;\theta)$，并且使用 θ^- 初始化目标网络 $Q(s,a;\theta^-)$
对于所有 *episode = 1* **转到** E **进行**
    初始化车辆信息矩阵，第一行为全 0，其余元素全为-1
    对于所有 i *= 1* **转到** T **进行**
        如果 在 $[0,1]$ 区间内随机取出的一个实数小于 ϵ **则**
            随机选择动作 a
        否则
            $a = \arg\max_a Q(s,a;\theta)$，其中 Q 值经过掩膜处理
        如果 $i = T$ **则**
            在环境中应用动作 a 并得到最终奖励 r_m 和下一个状态 s'
            $r = r_m$
        否则
            在环境中应用动作 a 并得到即时奖励 r_b 和下一个状态 s'
            $r = r_b$
        将四元组 (s,a,r,s') 添加到经验池 D 中，同时如果 $|D| > M$，删去经验池中最旧的经验
        $s = s', i = i + 1$
        如果 $|D| > M$ 且 $i > tp$ **则**
            从经验池 D 中随机取出 B 个四元组 (s,a,r,s')
            通过式(3.4)计算损失函数 L
            使用 Adam 反向传播通过 L 更新 θ
            更新目标网络参数 $\theta^- = \alpha\theta + (1-\alpha)\theta^-$
            更新 $\epsilon = \epsilon - \beta$

3.4 数据集与实验设置

3.4.1 真实世界的时刻表与 DRL-TSBC 生成的时刻表

本文使用了青岛市的 59 线与 85 线的发车时刻表与车辆行驶信息作为真实世界的测试数据。同时还使用了 DRL-TSBC 算法生成的 A 市 211 线的时刻表作为测试数据。

训练所用的数据集包括当前线路上的两个方向的公交发车时刻表 T_1 和 T_2，以及根据交通大数据统计出的每个小时公交从首站到末站所需要的时间。青岛市 59 线与 85 线的时刻表为目前正在实际运营中使用的时刻表，而 A 市 211 线的时刻表是 DRL-TSBC 算法生成的时刻表。各个线路的时刻表信息和约束条件如表3-1所示。

表 3-1 公交时刻表信息

线路	59	85	211
首班车时间	6:00	5:30	6:00
末班车时间	20:00	23:00	23:00
时间表中的总发车次数	104	170	136
最大工作时间 W_{max}（h）	16	16	10
最大驾驶时间 H_{max}（h）	13	13	8
最大行程次数 N_t	16	16	12
最短休息时间 r_{min}（min）	3	3	3

3.4.2 实验设置

本文搭建了一个可用于模拟公交车在线路上往返运行的仿真环境，并通过此环境进行训练与测试。DRL-BSA 的奖励函数中的最终奖励与即时奖励权重如表3-2所示，超参数如表3-3所示。D3QN 的网络结构如表3-4所示，其中 N 是预估的决策时最大可用的车辆数量，对于青岛 59、青岛 85 和 A 市 211 线来说，N 的取值分别为 12，16，10。

所有测试均在笔记本电脑上完成，CPU 型号为 AMD Ryzen 9 7945HX，GPU 型号为 NVIDIA RTX 4060 Laptop，内存大小为 32GB。DRL-BSA 使用 Python 中的 Pytorch 库实现。在青岛公交线路的对比中，采用的是开源数据集中的排班方案。

表 3-2 DRL-BSA 的奖励函数权重

权重	w_1^1	w_2^1	w_3^1	w_4^1	w_5^1	w_1^2	w_2^2	w_3^2	w_4^2	w_5^2
值	15	0.6	1.2	0.6	0.3	0.4	1	1	0.25	1

表 3-3 D3QN 网络的参数

参数	值
学习率	0.0001
权重初始化方法	正态分布初始化
批次大小 B	128
折现系数 γ	0.99
经验池大小 M	2^{17}
预训练步数 tp	2^{10}
ϵ	1.0
衰减率 β	0.0001
最大模拟次数 E	500
目标网络更新率 α	0.0001

表 3-4 D3QN 网络结构

网络名称	层数	输出节点数	激活函数
状态网络	1	80	ReLU
	2	256	无
V1 网络	1	80	ReLU
	2	1	无
A1 网络	1	80	ReLU
	2	N	无
V2 网络	1	80	ReLU
	2	1	无
A2 网络	1	80	ReLU
	2	N	无

3.5 离线车辆调度实验结果

DLR-BSA 的离线排班结果如表3-5所示，可以看到在青岛 59 线中，DRL-BSA 比人工方案使用更少的车辆就完成了发车时刻表，并且所有车辆的发车次数均为偶数次，而在青岛 85 线中，DRL-BSA 比人工方案使用的车辆数多一辆，这是由于人工方案并不满足约束条件中的一辆车最大行程数量为 16 次，其存在一辆车完成了 20 次行程的情况。而使用 DRL-TSBC 生成发车时刻表的 A 市 211 线，由于没有人工方案的发车时刻表，无法进行对比，但是同样可以看到 DRL-BSA 在进行车辆调度时，实现了所有车辆的行程数均为偶数次。

表 3-5 DRL-BSA 的离线车辆调度结果与人工方案的对比

线路	方法	N_u	N_o
青岛 59	人工方案	11	0
	DRL-BSA	8	0
青岛 85	人工方案	11	0
	DRL-BSA	12	0
A 市 211	人工方案	-	-
	DRL-BSA	15	0

DRL-BSA 生成的青岛 85 线的车辆调度结果如图3-4所示，每一行代表一辆车辆在一天内的所有行程，每一个矩形代表了一次行程，矩形的长度表示此次行程的时间，矩形越长代表行程时间越长。可以看到在 DRL-BSA 生成发车计划中，所有车辆的发车次数均为偶数。从离线车辆调度结果中可以看出，DRL-BSA 可以对车辆进行高效的排班，以完成发车时刻表。

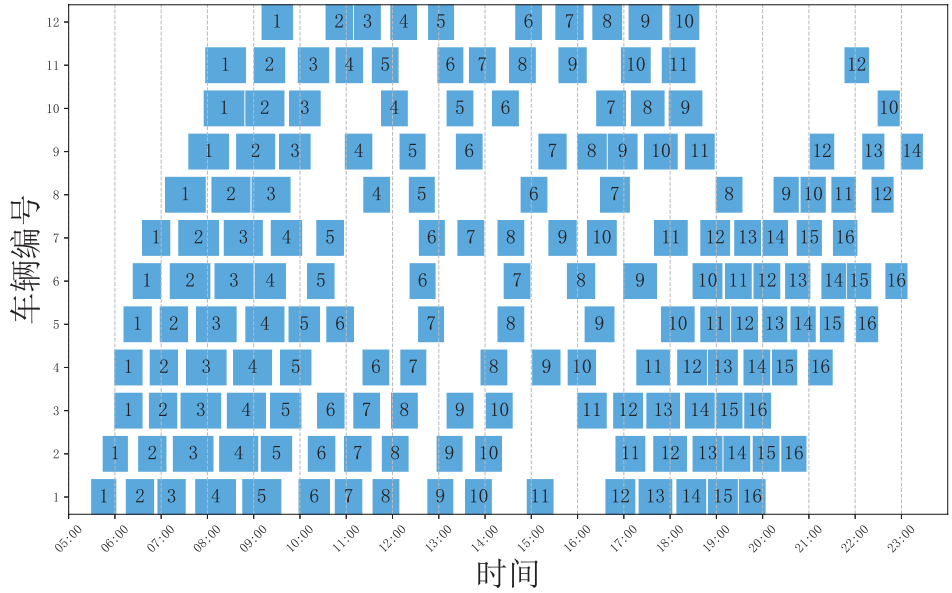

图 3-4 青岛 85 线的车辆调度结果

3.6 在线车辆调度实验结果

在公交运营过程中，公交车完成行程所需要的时间可能受到异常天气、交通事故、交通拥堵等不确定因素的影响而增加。如果继续使用离线车辆调度的调度方案，则会出现由于车辆无法即时完成行程而导致下一个计划行程无法进行。在发生不确定事件后，DRL-BSA 可以根据实时的道路情况，从突发事件出现的时刻开始重新安排车辆，即进行在线车辆调度。

为了对模拟的突发情况进行模拟，假设在早上 10:00 至 11:59 这段时间里，公交车完成一次行程的时间明显增加，这将会导致一些车辆无法即时返回出发点，而无法按时开始下一次行程。对于这种情况，一般人工方案会增加车辆数量，或是手动进行频繁的车辆调整，而这种重调整一般需要对后续的所有发车计划进行改动，这将会大大增加人力成本。

图3-5为没有发生交通拥堵情况下的车辆调度结果，图3-6则为交通拥堵发生后 DRL-TSBC 的在线排班结果。在图3-5的上午 10:00 至 11:59 发生了交通拥堵，导致车辆在这段时间内完成行程所需的时间增加，即图3-6中的拥堵时段。行驶时间的延长会导致车辆无法按照计划完成原有的发车任务。例如原本的 2 号车的第六次行程的时间是 10:11-10:45，第七次行程的开始时间是 10:58，如图3-5绿色标记所示。而当发生拥堵后，2 号车的第六次行程的时间变为 10:11-11:11，这导致了原计划中应于 10:58 开始的第七次行程无法按时开始。

为了应对交通拥堵带来的变化，DRL-BSA 会根据实时的道路情况，重新进行

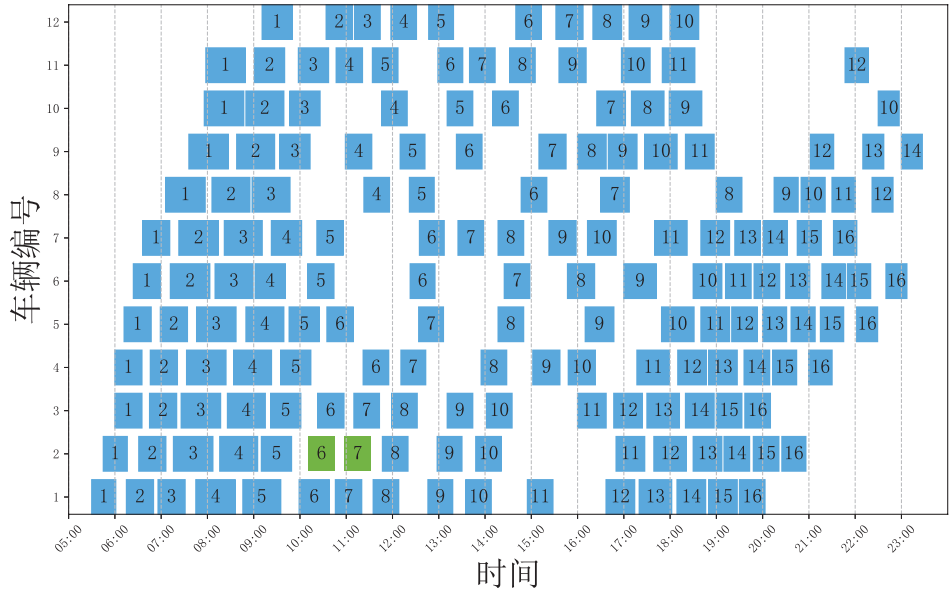

图 3-5 原始的排班结果

车辆调度。为了完成原本 2 号车的第七次行程，DRL-BSA 选择了 10 号车的第四次行程来进行替代，如图3-6中的黄色标记所示。这样就完成了对原本计划中无法按时开始的行程的替代。类似的，其他车辆的行程也会根据实时情况进行调整，以保证不会有车辆因为无法按时返回而导致发车计划失效。对比图3-5和图3-6可以看到，在发生了交通拥堵后，DRL-BSA 可以在不增加车辆数量的情况下，完成车辆的在线重新排班。

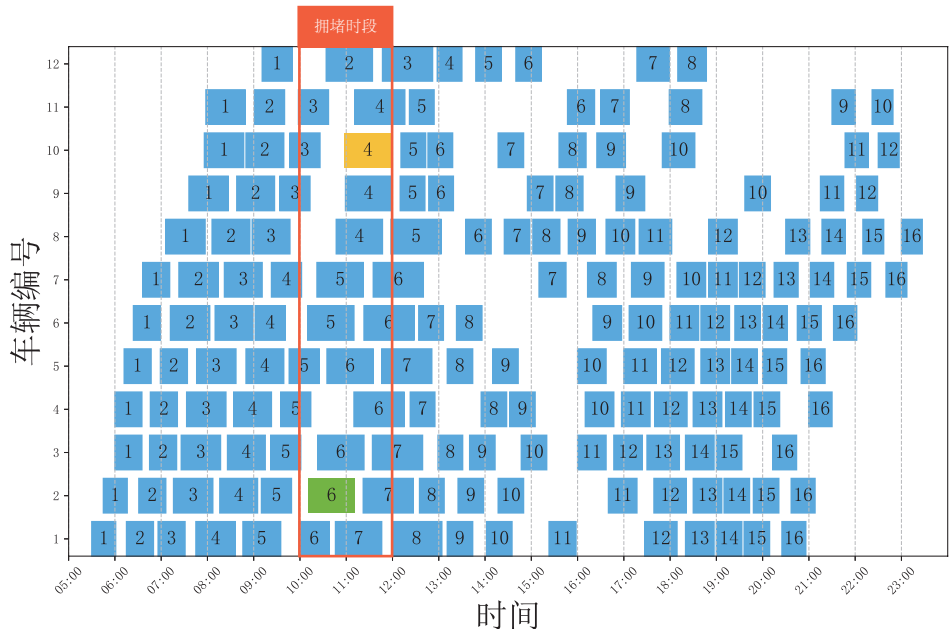

图 3-6 交通拥堵情况下 DRL-BSA 对于车辆的在线重新排班结果

3.7 消融实验

为了验证 DRL-BSA 中使用的各个方法的有效性，进行以下消融实验。

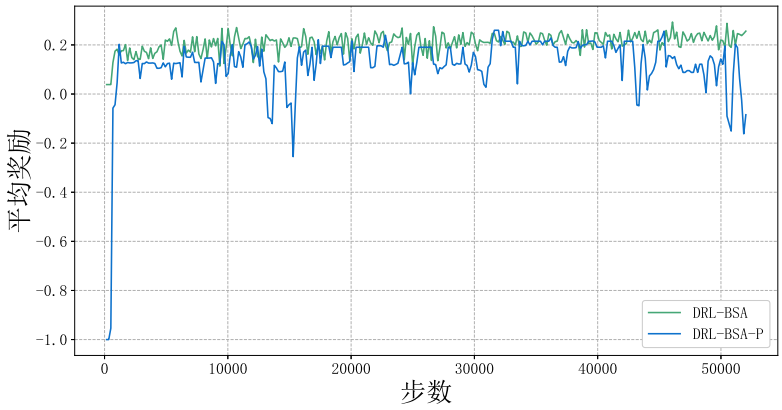

图 3-7 DRL-BSA 和 DRL-BSA-P 的对比

首先验证 DRL-BSA 中动作掩膜的有效性。在 DRL-BSA 中，为了避免智能体在进行选择时选择到不可用的车辆，即 V_u 中的车辆，DRL-BSA 在 D3QN 给出了所有动作的 Q 值以后，将代表不可用车辆对应的 Q 值设为了负无穷，以避免智能体选择到不可用车辆。为了验证这个方法的有效性，本文设计了 DRL-BSA-P，在 DRL-BSA-P 中，如果选择到了不可用车辆，当前时刻的奖励值将会变为-1，如果没有选择到不可用车辆奖励函数则会保持不变。如果智能体选择了一个不可用车辆，则相当于此次生成车辆调度计划失败，随后从头开始重新进行生成。DRL-BSA 与 DRL-BSA-P 在训练过程中的平均奖励值对比结果如图3-7所示，可以看到 DRL-BSA-P 的平均奖励函数值要明显低于 DRL-BSA，且并不收敛，这说明了动作掩膜的有效性。出现这种现象的原因是若在选择不可用车辆后引入惩罚，智能体将会过分重视避免选择不可用车辆，而忽略了其他的奖励信息，导致最终奖励不收敛。

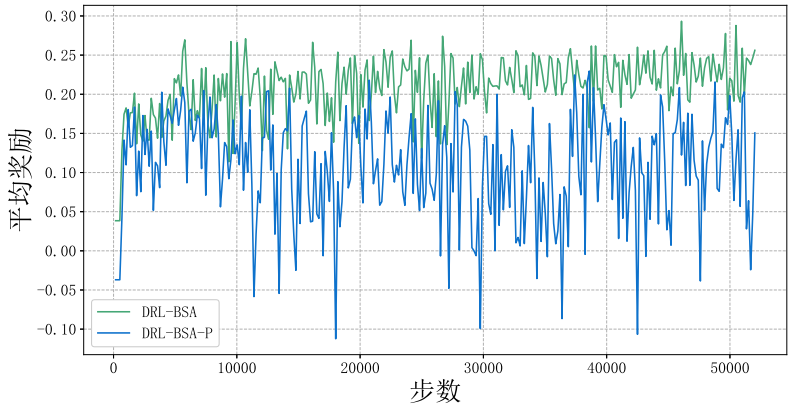

图 3-8 DRL-BSA 和 DRL-BSA-N 的对比

其次为验证 DRL-BSA 使用的 D3QN 的有效性，本文设计了使用 DQN 作为深度强化学习网络的 DRL-BSA-N。DRL-BSA 与 DRL-BSA-N 在训练过程中的平均奖励值的对比结果如图3-8所示，可以看到使用 D3QN 的 DRL-BSA 对比使用 DQN 的 DRL-BSA-N 平均奖励更高，同时也更加稳定。此外，DRL-BSA-N 最终得到的目标网络与主网络之间的均方误差值约为 0.58326，而 DRL-BSA 的目标网络与主网络之间的均方误差值约为 0.005151，差距约为两个数量级。这说明 D3QN 相比于 DQN 对 Q 值的估计更加准确，同时其有效减少了 Q 值高估带来的问题，使 DRL-BSA 的收敛曲线更加稳定。

3.8 本章小结

在本章中，详细介绍了基于深度强化学习的公交车辆调度算法，本文将公交车辆调度问题建模为马尔可夫决策过程，将车辆的行程数量、剩余驾驶时间、剩余工作时间和休息时间作为状态。同时设计了一个包含即时奖励和最终奖励的奖励函数以引导智能体做出决策。在动作选择时，使用了一个掩膜层来避免选择到不可用的车辆。本文使用 D3QN 在每一个决策时间点选择一辆车辆完成发车任务，以解决传统 DQN 在该问题上遇到的 Q 值高估问题。

对 DRL-BSA 的离线车辆调度与在线车辆调度性能进行的测试结果表明，在与人工方案进行对比时，DRL-BSA 的排班方案能够使用更少的车辆数量完成时刻表，同时较好的保证了车辆的行程数量均为偶数次。对 DRL-TSBC 所生成的时刻表进行离线排班的结果也说明 DRL-BSA 可应用在 DRL-TSBC 生成的公交时刻表的后续排班的流程中。在线测试表明，当发生交通拥堵等突发情况时，DRL-BSA 能够根据实时情况进行车辆的在线重新排班，而不需要增加车辆数量。消融实验结果表明，DRL-BSA 中使用的动作掩膜和 D3QN 网络对于 DRL-BSA 的性能有着显著的提升。

第 4 章 总结与展望

4.1 工作总结

在本文中，以交通大数据为基础，对公交时刻表排班问题与公交车辆调度问题进行了研究。

针对公交时刻表排班问题，本文提出了基于深度强化学习的双向动态公交时刻表排班算法（DRL-TSBC），其将双向公交时刻表的实时排班问题建模为一个马尔可夫决策过程，通过对状态空间、动作空间和奖励函数的设计，以 DQN 做为深度强化学习网络，实现了双向公交时刻表的在线动态排班。在真实数据集上的实验结果表明，DRL-TSBC 算法在与现有的人工方案进行对比时，可以更有效地调度车辆，降低乘客的平均等待时间并保持上下行发车次数相等。在遇到客流量的突变时，DRL-TSBC 可以根据变化的客流做出相应的调整，并且仍然保持上下行发车次数相等。

此外针对公交车辆调度问题，本文提出了基于深度强化学习的公交车辆调度算法（DRL-BSA），其将公交车辆调度问题建模为一个马尔可夫决策过程，通过对状态空间、动作空间和奖励函数的设计，以 D3QN 做为深度强化学习网络，实现了公交车辆的在线排班。其在真实世界的已有时刻表上的车辆调度结果对比人工方案，可以更有效地调度车辆，在最少化所需车辆的情况下使得车辆的行程次数均为偶数次。在遇到交通拥堵的情况时，DRL-BSA 可以在不增加车辆数量的情况下进行在线调整，以应对交通状况的突变。

于此同时，DRL-BSA 也在 DRL-TSBC 生成的时刻表上进行了测试，实验结果表明 DRL-BSA 也可在 DRL-TSBC 生成的时刻表上进行有效的排班。本文设计的两个方法可以作为公交排班调度问题中时刻表排班和车辆调度的两个环节，实现了公交排班调度问题的完整解决方案。

4.2 研究展望

在未来的研究中，可以在本文实现的方法中增加更多的公交业务需求的约束，例如在公交时刻表排班中加入对于发班均匀度的控制方法，或是在公交车辆调度中加入最大等待时间的约束条件。

此外，公交时刻表排班和公交车辆调度作为公交排班调度的两个方面，现有

的方法均是对其分开进行处理，但实际上如果在进行公交时刻表排班时就考虑到公交车辆调度的问题，可能可以更好地降低成本并提升乘客的满意程度，同时也更加便于公交公司进行管理。未来的研究可以尝试使用深度强化学习方法，将公交时刻表排班问题与公交车辆调度问题进行联合建模，实现公交排班调度问题的一体化解决方案。

在完成公交车辆调度之后，公交公司还需为车辆分配驾驶员，这也是一个重要的排班问题。未来的研究也可尝试使用深度强化学习方法，对公交驾驶员分配问题进行研究，真正实现从原始的交通大数据到完整可用的运营方案的全流程研究。

参考文献

[1] GAO Y, ZHU J. Characteristics, Impacts and Trends of Urban Transportation[J]. Encyclopedia, 2021, 1(1): 1167–1182.

[2] WANG C, GUO C, ZUO X. Solving multi-depot electric vehicle scheduling problem by column generation and genetic algorithm[J]. Applied Soft Computing, 2021, 112: 107774.

[3] PARBO J, NIELSEN O A, PRATO C G. User perspectives in public transport timetable optimisation[J]. Transportation Research Part C: Emerging Technologies, 2014, 48: 269–284.

[4] ANON. Cost, production and efficiency in local bus industry: An empirical analysis for the bus system of Santiago[J]. Transportation Research Part A: Policy and Practice, 2018, 108: 1–11.

[5] FRELING R, HUISMAN D, WAGELMANS A P M. Applying an Integrated Approach to Vehicle and Crew Scheduling in Practice[M] // VOSS S, DADUNA J R. Computer-Aided Scheduling of Public Transport. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001: 73–90.

[6] CEDER A, TAL O. Timetable synchronization for buses[G] // Computer-Aided Transit Scheduling: Proceedings, Cambridge, MA, USA, August 1997. [S.l.]: Springer, 1999: 245–258.

[7] LUHUA S, YIN H, XINKAI J. Study on method of bus service frequency optimal ModelBased on genetic algorithm[J]. Procedia Environmental Sciences, 2011, 10: 869–874.

[8] GKIOTSALITIS K, ALESIANI F. Robust timetable optimization for bus lines subject to resource and regulatory constraints[J]. Transportation Research Part E: Logistics and Transportation Review, 2019, 128: 30–51.

[9] TANG J, YANG Y, HAO W, et al. A data-driven timetable optimization of urban bus line based on multi-objective genetic algorithm[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 22(4): 2417–2429.

[10] YU B, YANG Z, YAO J. Genetic algorithm for bus frequency optimization[J]. Journal of Transportation Engineering, 2010, 136(6): 576–583.

[11] CEDER A. Bus frequency determination using passenger count data[J]. Transportation Research Part A: General, 1984, 18(5-6): 439–453.

[12] WIHARTIKO F, BUONO A, SILALAHI B. Integer programming model for optimizing bus timetable using genetic algorithm[J], 2017, 166(1): 012016.

[13] SHANG H-Y, HUANG H-J, WU W-X. Bus timetabling considering passenger satisfaction: An empirical study in Beijing[J]. Computers & Industrial Engineering, 2019, 135: 1155–1166.

[14] YAN H, CUI Z, CHEN X, et al. Distributed multiagent deep reinforcement learning for multi-line dynamic bus timetable optimization[J]. IEEE Transactions on Industrial Informatics, 2022, 19(1) : 469 – 479.

[15] LI J, DONG H, ZHAO X, et al. Practical bus timetable optimization method based on deep reinforcement learning[C] // 2022 4th International Academic Exchange Conference on Science and Technology Innovation (IAECST). 2022 : 581 – 587.

[16] ZHAO Y, CHEN G, MA H, et al. Dynamic Bus Holding Control Using Spatial-Temporal Data–A Deep Reinforcement Learning Approach[J], 2022 : 661 – 674.

[17] AI G, ZUO X, CHEN G, et al. Deep reinforcement learning based dynamic optimization of bus timetable[J]. Applied Soft Computing, 2022, 131 : 109752.

[18] LIU Y, ZUO X, AI G, et al. A reinforcement learning-based approach for online bus scheduling[J]. Knowledge-Based Systems, 2023, 271 : 110584.

[19] LIU Y, ZUO X, LI X, et al. A genetic algorithm with trip-adjustment strategy for multi-depot electric bus scheduling problems[J]. Engineering Optimization, 2023 : 1 – 20.

[20] CHEN X, KONG Y, DANG L, et al. Exact and metaheuristic approaches for a bi-objective school bus scheduling problem[J]. PloS one, 2015, 10(7) : e0132600.

[21] BOYER V, IBARRA-ROJAS O J, RÍOS-SOLÍS Y Á. Vehicle and crew scheduling for flexible bus transportation systems[J]. Transportation Research Part B: Methodological, 2018, 112 : 216 – 229.

[22] JANOVEC M, KOHÁNI M. Exact approach to the electric bus fleet scheduling[J]. Transportation Research Procedia, 2019, 40 : 1380 – 1387.

[23] GKIOTSALITIS K, ILIOPOULOU C, KEPAPTSOGLOU K. An exact approach for the multi-depot electric bus scheduling problem with time windows[J]. European Journal of Operational Research, 2023, 306(1) : 189 – 206.

[24] WEN M, LINDE E, ROPKE S, et al. An adaptive large neighborhood search heuristic for the electric vehicle scheduling problem[J]. Computers & Operations Research, 2016, 76 : 73 – 83.

[25] YAO E, LIU T, LU T, et al. Optimization of electric vehicle scheduling with multiple vehicle types in public transport[J]. Sustainable Cities and Society, 2020, 52 : 101862.

[26] YANG X, QI Y. Research on optimization of multi-objective regional public transportation scheduling[J]. Algorithms, 2021, 14(4) : 108.

[27] SHEN Y, XU J, LI J. A probabilistic model for vehicle scheduling based on stochastic trip times[J]. Transportation Research Part B: Methodological, 2016, 85 : 19 – 31.

[28] TANG X, LIN X, HE F. Robust scheduling strategies of electric buses under stochastic traffic conditions[J]. Transportation Research Part C: Emerging Technologies, 2019, 105 : 163 – 182.

[29] WANG C, SHI H, ZUO X. A multi-objective genetic algorithm based approach for dynamical bus vehicles scheduling under traffic congestion[J]. Swarm and Evolutionary Computation, 2020, 54 : 100667.

[30] MATTHEWS R A. The science of murphy’s law[J]. Scientific American, 1997, 276(4) : 88 – 91.

[31] GAO P, ZUO X, BIAN Q, et al. A memetic algorithm to optimize bus timetable with unequal time intervals[C] // Proceedings of the genetic and evolutionary computation conference companion. 2019 : 1336 – 1344.

[32] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning[J]. nature, 2015, 518(7540) : 529 – 533.

[33] WANG Z, SCHAUL T, HESSEL M, et al. Dueling network architectures for deep reinforcement learning[C] // International conference on machine learning. 2016 : 1995 – 2003.

[34] VAN HASSELT H, GUEZ A, SILVER D. Deep reinforcement learning with double q-learning[C] // Proceedings of the AAAI conference on artificial intelligence : Vol 30. 2016.

[35] FUJIMOTO S, HOOF H, MEGER D. Addressing function approximation error in actor-critic methods[C] // International conference on machine learning. 2018 : 1587 – 1596.

相关的科研成果目录

1. 在投论文

[1] XIE J, Lin Z, Yin J, et al. Deep Reinforcement Learning Based Dynamic Bus Timetable Scheduling with Bidirectional Constraints[C] // Big Data and Social Computing: 9th China National Conference, BDSC 2024, Harbin, China, August 8–10, 2024.

致谢

时光匆匆，转眼间大学本科的四年已经接近尾声，回首这半年来对于毕业论文的研究，我的心中感慨万千。在此，我要向所有在我完成毕业论文过程中给予我帮助的人表示衷心的感谢。

首先需要感谢一直默默支持着我的父母，在我毕业论文研究的过程中给予的精神上的鼓励和帮助，让我在最迷茫和最焦虑的时候能够坚持下来。感谢陈翔老师从大一以来对我的指导和帮助，让我的大学生活更加充实，也让我能够在我感兴趣的方向上完成这份毕业论文。感谢王玺钧老师在我毕业论文写作过程中给予的指导和帮助，让我能够在论文写作的过程中不断学习到新的知识与技巧。感谢尹杰丽师姐在整个研究过程中给予的支持，为我提供了研究方向的启发和帮助。感谢实验室的学长学姐在我刚接触大数据时给予的帮助，特别感谢林卓学姐给予的灵感和在代码测试上提供的帮助。

最后，对大学四年以来所有关心和帮助我的人表示由衷的感谢。

姓名　谢嘉昊

2024 年 4 月 28 日