MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 33, NO. 1, 2026
journal homepage: www.mais-journal.ru

ARTIFICIAL INTELLIGENCE

Reinforcement Learning for Urban Public Transport Driver Scheduling

S. V. Goncharov[1], I. S. Vojteshenko[1]

DOI: 10.18255/1818-1015-2026-1-30-47

[1]Belarusian State University, Minsk, Belarus

MSC2020: 68T05
Research article
Full text in Russian

Received February 5, 2026
Revised February 21, 2026
Accepted February 25, 2026

The article examines the application of deep reinforcement learning methods to solve the problem of automated scheduling of driver shifts for urban passenger transport. The Crew Scheduling Problem belongs to the class of NP-hard combinatorial optimization problems and is characterized by a multitude of complex constraints related to labor legislation and the operational specifics of the transport network. The problem formulation, considering route changes, is described. The problem is formalized as a Markov Decision Process, taking into account specific constraints of the transport industry: maximum working hours, lunch breaks, and minimum rest time between trips. The state space is formulated, including features of control stops, the current trip, and candidate shifts. A mechanism for prioritized selection of candidate shifts is described to reduce the dimensionality of the action space. A multi-component reward function is described, considering the number of shifts utilized, deadhead travel time, and driver utilization. The agent's architecture is implemented based on the Actor-Critic method with the Proximal Policy Optimization algorithm. An experimental study was conducted on real data from the transport network of the city of Yaroslavl, including 6 routes and 974 trips. A comparative analysis was conducted with alternative methods: DQN, REINFORCE, and a heuristic approach represented by a greedy algorithm. The comparative analysis of the results demonstrated the superiority of the PPO algorithm over the other approaches. As a result of the study, it was concluded that reinforcement learning methods can be used to solve transportation optimization problems.

Keywords: reinforcement learning; deep learning; PPO; crew scheduling problem; schedule; public transport; neural networks

INFORMATION ABOUT THE AUTHORS

Goncharov, Sergei V. (corresponding author)	ORCID iD: 0009-0008-5126-4344. E-mail: goncharov.sergei.21@gmail.com Student
Vojteshenko, Iosif S.	ORCID iD: 0000-0002-0134-1793. E-mail: voit@bsu.by PhD, Associate Professor

For citation: S. V. Goncharov and I. S. Vojteshenko, "Reinforcement learning for urban public transport driver scheduling", *Modeling and Analysis of Information Systems*, vol. 33, no. 1, pp. 30–47, 2026. DOI: 10.18255/1818-1015-2026-1-30-47.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 33, № 1, 2026
сайт журнала: www.mais-journal.ru

Применение обучения с подкреплением в задаче составления графиков смен водителей городского общественного транспорта

С. В. Гончаров[1], И. С. Войтешенко[1]

DOI: 10.18255/1818-1015-2026-1-30-47

[1]Белорусский государственный университет, Минск, Беларусь

УДК 004.032.26+656.072
Научная статья
Полный текст на русском языке

Получена 5 февраля 2026 г.
После доработки 21 февраля 2026 г.
Принята к публикации 25 февраля 2026 г.

В статье рассматривается применение методов глубокого обучения с подкреплением для решения задачи автоматизированного составления графиков водительских смен городского пассажирского транспорта. Задача составления графиков смен (Crew Scheduling Problem) относится к классу NP-трудных задач комбинаторной оптимизации и характеризуется множеством сложных ограничений, связанных с трудовым законодательством и операционными особенностями транспортной сети. Описана постановка задачи с учётом смены маршрутов. Предложена формализация задачи в виде марковского процесса принятия решений с учётом специфических ограничений транспортной отрасли: максимального рабочего времени, обеденных перерывов и минимального времени отдыха между рейсами. Сформулировано пространство состояний, включающее признаки контрольных остановок, текущего рейса и смен-кандидатов. Описан механизм приоритетного отбора смен-кандидатов для снижения размерности пространства действий. Описана многокомпонентная функция награды, учитывающая число задействованных смен, время холостых поездок и утилизацию водителей. Архитектура агента реализована на основе метода Actor-Critic с алгоритмом Proximal Policy Optimization. Экспериментальное исследование проведено на реальных данных транспортной сети города Ярославль, включающей 6 маршрутов и 974 рейса. Проведён сравнительный анализ с альтернативными методами: DQN, REINFORCE и эвристическим подходом, представленного жадным алгоритмом. Сравнительный анализ результатов показал превосходство алгоритма PPO над другими подходами. В результате исследования сделан вывод о возможности использования методов обучения с подкреплением для решения задач транспортной оптимизации.

Ключевые слова: обучение с подкреплением; глубокое обучение; PPO; составление графиков; расписание; общественный транспорт; нейронные сети

ИНФОРМАЦИЯ ОБ АВТОРАХ

Гончаров, Сергей Витальевич (автор для корреспонденции)	ORCID iD: 0009-0008-5126-4344. E-mail: goncharov.sergei.21@gmail.com Студент
Войтешенко, Иосиф Станиславович	ORCID iD: 0000-0002-0134-1793. E-mail: voit@bsu.by Канд. тех. наук, доцент

Для цитирования: S. V. Goncharov and I. S. Vojteshenko, “Reinforcement learning for urban public transport driver scheduling”, *Modeling and Analysis of Information Systems*, vol. 33, no. 1, pp. 30–47, 2026. DOI: 10.18255/1818-1015-2026-1-30-47.

Введение

Эффективное управление городским пассажирским транспортом является одной из ключевых задач современной транспортной логистики. Задача составления графиков работы водителей (Crew Scheduling Problem, CSP) занимает важное место в операционном планировании транспортных предприятий, поскольку затраты на персонал составляют значительную долю операционных расходов [1].

Задача CSP относится к классу NP-трудных задач комбинаторной оптимизации и характеризуется множеством сложных ограничений, связанных с трудовым законодательством, требованиями к отдыху водителей и операционными особенностями транспортной сети. Традиционные методы точной оптимизации, такие как целочисленное линейное программирование и методы ветвей и границ, часто оказываются вычислительно затратными при масштабировании задачи [1]. Эвристические подходы, хотя и обеспечивают приемлемое время работы, не гарантируют качества получаемых решений.

В последние годы методы машинного обучения, в частности обучение с подкреплением (Reinforcement Learning, RL), демонстрируют значительный потенциал в решении задач комбинаторной оптимизации [2]. Ключевым преимуществом RL-подходов является способность адаптироваться к сложным нелинейным ограничениям и формировать эффективные стратегии принятия решений на основе взаимодействия со средой.

Целью настоящей работы является описание и экспериментальная проверка подхода применения глубокого обучения с подкреплением для автоматизированного составления графиков водительских смен в задаче CSP с минимизацией операционных расходов. Предлагаемый подход ориентирован на построение конструктивной стратегии назначения смен на рейсы при учёте прикладных ограничений режима труда и затрат, связанных с холостыми поездками.

Статья организована следующим образом. В разделе 1 формализуется задача CSP, вводятся ограничения и критерии качества решения. В разделе 2 приводится обзор существующих подходов к решению CSP, включая точные методы, метаэвристики и методы обучения с подкреплением. В разделе 3 описывается модель и реализация RL-агента, включая пространство состояний и действий, функцию награды и используемые архитектурные решения. В разделе 4 представлены экспериментальные результаты на основе реальных данных и сравнение с другими подходами. В разделе 5 обсуждаются возможности практического применения и внедрения предлагаемого подхода.

1. Постановка задачи

Пусть задано множество автобусных маршрутов, состоящих из рейсов в двух направлениях. Каждый маршрут имеет две контрольные остановки, которые являются точками отправления и прибытия рейсов, а также местами отдыха между рейсами. Для каждой контрольной остановки задано расписание, содержащее времена отправления, где каждому времени отправления соответствует один рейс. Рейс представляет собой путь от одной контрольной остановки до другой в рамках маршрута, выполняемый в определённый промежуток времени.

Пусть задача включает n контрольных остановок. Расписание k-й контрольной остановки обозначим как $T_k = \{t_1^k, t_2^k, t_3^k, \ldots, t_{N_k}^k\}$, где N_k — число времён отправления для k-й контрольной остановки. Расписания всех n контрольных остановок объединяются в единое расписание T, в котором все времена отправления упорядочены в хронологическом порядке.

Пусть время отправления i-го рейса водительской смены v равно d_i^v, а время прибытия — a_i^v. Тогда время в пути для данного рейса составляет $dur_i^v = a_i^v - d_i^v$. После завершения рейса в рамках смены v может быть выполнен следующий рейс — либо на текущем маршруте, либо на другом. При продолжении работы на текущем маршруте время между прибытием с рейса i и отправлением на следующий рейс $(i+1)$ является временем отдыха: $r_i^v = d_{i+1}^v - a_i^v$. В противном случае необходима

холостая поездка до контрольной остановки другого маршрута. Обозначим время холостой поездки после рейса i как k_i^v; если холостая поездка не требуется, то $k_i^v = 0$.

Ограничения задачи формулируются следующим образом:

1. В рамках смены v текущий рейс i должен быть завершён прежде, чем начнётся следующий рейс $(i + 1)$.
2. Время отдыха после завершения рейса i не может быть меньше минимального времени отдыха $R_{\min}$, т. е. $r_i^v \geqslant R_{\min}$.
3. Если после завершения рейса i рейс $(i + 1)$ выполняется на другом маршруте, то должно выполняться условие: $r_{\min} + k_i^v \leqslant d_{i+1}^v - a_i^v$.
4. Суммарное рабочее время в рамках смены не должно превышать $W_{\max}$.
5. После работы не менее $L_{\min}$ минут в рамках смены должен быть предоставлен обеденный перерыв продолжительностью не менее L_{dur}.
6. Все времена рейсы в объединённом расписании T должны быть выполнены.

Цели задачи состоят в минимизации:

1. Общего числа задействованных смен N_{shifts}.
2. Суммарного времени холостых поездок $T_{deadhead} = \sum_{v=1}^{N_{shifts}} \sum_{i=1}^{N^v} k_i^v$, где N^v — число рейсов смены v.

Задача формализуется как последовательность решений: для каждого рейса из объединённого расписания T необходимо выбрать водительскую смену, которая должна отвечать за выполнение этого рейса, или завести новую, если текущего числа смен недостаточно. В результате получается набор водительских смен, выполняющих все рейсы расписания.

2. Обзор литературных источников

Существующие подходы можно разделить на точные методы математического программирования, эвристические и метаэвристические методы и методы машинного обучения, в частности RL. Выбор класса методов, как правило, определяется компромиссом между гарантией оптимальности/оценкой качества решения и вычислительными затратами при масштабировании, а также сложностью внедрения с учётом прикладных ограничений.

2.1. Точные методы оптимизации

Классическая формулировка CSP в виде задачи покрытия множества (Set Covering Problem) или разбиения множества (Set Partitioning Problem) остаётся стандартом в отрасли [1]. В такой постановке каждому рейсу соответствует строка матрицы ограничений, а столбцам соответствуют допустимые смены, каждая из которых покрывает некоторое подмножество рейсов. Требуется выбрать подмножество смен минимальной стоимости, покрывающее все рейсы.

Ключевое ограничение данной формулировки заключается в том, что число потенциально допустимых смен обычно экспоненциально по размеру расписания, поэтому прямое перечисление столбцов становится неосуществимым уже на экземплярах промышленного масштаба. Это приводит к необходимости тщательно проектировать правила генерации смен и применять декомпозиционные техники, иначе даже современные решатели задач смешанного целочисленного программирования (Mixed Integer Programming, MIP) оказываются перегружены объёмом модели и временем поиска оптимума. Практический успех ранних систем демонстрирует экономический эффект точных постановок, однако он также подчёркивает их высокую зависимость от качества математической модели и корректной интеграции «предметных» ограничений в линейную структуру [3].

Для преодоления проблемы огромного числа столбцов применяется метод генерации столбцов (Column Generation) [4]. Его преимущество состоит в том, что модель решается в ограниченном пространстве переменных, а затем итеративно добавляются «наиболее перспективные» смены из подзадачи ценообразования. Однако область применимости данного метода определяется структурой

подзадачи: при усложнении ограничений (ресурсные ограничения на длительность смены, перерывы, минимальный отдых, зависимость перегонов от времени суток и т. п.) подзадача часто превращается в вариант кратчайшего пути с ресурсными ограничениями, что усложняет реализацию и может существенно замедлять итерации. Кроме того, на практике нередко требуется стабилизация двойственных оценок и специальные эвристики, поскольку иначе число итераций генерации столбцов может быть велико даже при умеренных размерах исходного расписания.

Для получения целочисленного решения метод генерации столбцов комбинируется с методом ветвей и границ в схеме branch-and-price [5]. Такие подходы дают сильные оценки качества (нижние границы) и способны находить оптимальные решения для задач малого и среднего размера, что делает их подходящими для офлайн-планирования в случаях, когда требуется формальная оптимальность и существует время на расчёт. Вместе с тем branch-and-price является методологически и программно сложным: необходимо проектировать ветвление, совместимое с генерацией столбцов, контролировать вырождение и обеспечивать устойчивость решения релаксаций задач линейного программирования. При росте масштаба и усложнении ограничений гарантия нахождения оптимума в разумное время практически исчезает.

2.2. Эвристические и метаэвристические методы

Из-за вычислительной сложности точных методов значительное распространение получили эвристические и метаэвристические подходы, ориентированные на получение хороших (но не обязательно оптимальных) решений за приемлемое время.

Жадные алгоритмы и подходы на основе жадного рандомизированного адаптивного поиска (Greedy Randomized Adaptive Search Procedure, GRASP) быстро строят допустимые решения и потому полезны как для оперативного планирования, так и для генерации начальных решений в гибридных схемах [6]. Их фундаментальное ограничение связано с локально выгодными назначениями (например, выбор смены с лучшим текущим показателем), которые могут ухудшать глобальную связность расписания и приводить к росту числа смен или времени холостых перегонов на последующих шагах, поскольку алгоритм не учитывает долгосрочные последствия выбора.

Генетические алгоритмы (Genetic Algorithms, GA) применяются для многокритериальной оптимизации графиков и позволяют естественным образом учитывать несколько конфликтующих целей и предпочтения персонала [7]. При этом применимость GA на практике во многом определяется способом кодирования смен и механизмами обеспечения допустимости: без тщательно настроенных штрафов за нарушения ограничений алгоритм может тратить значительную долю вычислений на недопустимые решения. Кроме того, качество результатов чувствительно к настройке параметров (размер популяции, оператор скрещивания, мутация, критерии отбора), что усложняет переносимость метода между различными транспортными сетями.

Метод запретов (Tabu Search) эффективен как средство локального улучшения за счёт механизмов памяти, позволяющих избегать циклов и выходить из локальных оптимумов [8]. Его ограничения связаны с зависимостью от выбора окрестности (наборов локальных преобразований расписания) и параметров tabu-списка: слишком агрессивная диверсификация может приводить к «шумному» поиску, а слишком слабая — к преждевременной сходимости. На сложных моделях CSP также возрастает стоимость проверки допустимости каждого локального шага, что может нивелировать преимущества метода по времени.

Имитация отжига (Simulated Annealing) вводит контролируемое принятие ухудшающих шагов и тем самым помогает преодолевать локальные экстремумы [9]. Однако качество и время работы существенно зависят от режима охлаждения и критериев остановки; при «медленном» охлаждении алгоритм становится вычислительно тяжёлым, а при «быстром» может давать нестабильное качество без возможности формальной оценки близости к оптимуму.

Алгоритмы оптимизации муравьиной колонией (ACO) хорошо сочетаются с сетевыми представлениями задач и показали эффективность в задачах маршрутизации [10]. Для CSP их применение обычно требует специального графового моделирования допустимых стыковок рейсов и аккуратного выбора эвристической информации; иначе метод может сталкиваться с проблемами стагнации (концентрация вероятности на ранних траекториях) и ухудшением масштабируемости при росте числа рейсов и вариантов состыковок.

Коммерческие системы (TRAPEZE, HASTUS, TRACS) интегрируют математические модели оптимизации с эвристиками и предоставляют интерактивные инструменты для планировщиков [11]. Их практическое преимущество заключается в зрелости и ориентированности на внедрение, однако ограничения часто связаны с высокой стоимостью владения, закрытостью алгоритмических компонентов и сложностью глубокой адаптации под нестандартные локальные ограничения, что снижает воспроизводимость результатов и гибкость научного сравнения.

2.3. Методы машинного обучения

Методы обучения с подкреплением рассматриваются как способ автоматизировать построение конструктивных эвристик для комбинаторной оптимизации и адаптироваться к сложным нелинейным взаимодействиям ограничений [2]. В отличие от традиционных эвристик, RL-агент потенциально может переиспользовать выученную стратегию на новых экземплярах задачи, а также дообучаться при изменении характеристик сети и расписания.

Ограничения RL-подходов во многом смещаются из области вычислительной сложности «решения модели» в область сложности «обучения политики»: для задач с большим пространством состояний и длинными эпизодами возрастает потребность в большом числе взаимодействий со средой, а качество существенно зависит от проектирования награды (reward shaping) и стабилизации обучения. В прикладных постановках CSP критичным становится вопрос обеспечения допустимости: если ограничения (например, отдых/обед/максимум рабочего времени) не встроены в среду как жёсткие, агент может «учиться» на штрафах и долго генерировать недопустимые решения; поэтому перспективны гибридные схемы, где RL комбинируется с методами ограничений/поиска для гарантии выполнимости [12].

Среди алгоритмов глубокого RL используются подходы Deep Q-Network (DQN) [13], REINFORCE [14] и Actor-Critic [15]. Для задач планирования с длинным горизонтом и разреженными сигналами награды более устойчивыми на практике часто оказываются методы семейства Proximal Policy Optimization (PPO), ограничивающие величину обновления политики и тем самым снижающие риск «коллапса» обучения [16]. Тем не менее применимость конкретного RL-алгоритма к CSP определяется не только выбором оптимизатора, но и представлением состояния, механизмом редукции/маскирования действий и способом учёта ограничений, поскольку именно эти компоненты задают достижимое качество и скорость сходимости.

Нейросетевые архитектуры внимания и Pointer Networks демонстрируют способность к обобщению в задачах маршрутизации и построения последовательностей решений [17, 18]. Их ограничение в контексте CSP связано с тем, что реальные ограничения труда и отдыха, а также перегоны и множественные «мягкие» критерии требуют либо усложнения архитектуры/среды, либо перехода к гибридным методам, иначе качество обобщения может резко ухудшаться при переносе между сетями.

Помимо задач управления и маршрутизации, RL применяется и к задачам персонального и ресурсного планирования, близким CSP. Так, Kenworthy и др. предложили гибридный подход NICE, в котором RL используется для аппроксимации сложной робастной цели и последующего построения устойчивых расписаний в задаче назначения пилотов на рейсы (flight crew scheduling) на основе целочисленного программирования [19]. Авторы показывают, что RL-информация может существенно улучшать устойчивость расписаний к сбоям при значительном снижении вычислительных

затрат по сравнению с прямой робастной MIP-формулировкой, что демонстрирует применимость RL-подходов в задачах crew scheduling и родственных постановках. Для близкой задачи составления графиков медперсонала (nurse scheduling) Nagayoshi и Tamaki рассматривают конструктивное построение расписания с последующей процедурой «исправлений» (work revision), где стратегия обменов смен обучается методами RL (в частности, Q-learning) [20]. В задаче nurse rostering предлагались схемы, в которых RL-компонент (например, DQN/мультиагентные варианты) встраивается в метаэвристический контур и управляет выбором операторов/окрестностей для улучшения расписания, что повышает качество поиска при большом числе мягких ограничений [21]. Наконец, для задач выездного обслуживания и полевых работ (technician routing and scheduling), объединяющих маршрутизацию и назначение ресурсов во времени, предложены постановки на основе глубокого обучения с подкреплением и алгоритмы для многопериодных вариантов задачи [22], что подчёркивает переносимость подходов на основе марковских процессов принятия решений и RL на широкий класс задач расписаний, включая CSP и родственные модели с ресурсными и временными ограничениями.

Liu и Zuo [23] предложили подход RL-MSA для многомаршрутного планирования автобусов (Multi Line Bus Scheduling Problem, MLBSP), который был использован в качестве исходной основы для построения марковского процесса принятия решений (Markov Decision Process, MDP) и общей схемы последовательного распределения рейсов между сменами в данной статье. В MLBSP в качестве точки принятия решения рассматривается каждое время отправления из объединённого расписания, а агент выбирает автобус для выполнения соответствующего рейса. При этом вводятся специальные признаки для контрольных остановок, текущего рейса и кандидатов-автобусов, а также механизм приоритизации кандидатов для ограничения размерности пространства действий. В настоящей работе решается иная по содержанию задача — CSP, где необходимо назначать на рейсы не автобусы, а водительские смены, причём допустимость решения определяется трудовыми ограничениями (минимальный отдых между рейсами, обеденный перерыв, ограничение суммарного рабочего времени смены) и операционными издержками, связанными с холостыми поездками и числом задействованных смен.

Для адаптации базового подхода к CSP изменена логика окружения и его динамика: среда моделирует последовательное формирование смен и контролирует выполнение ограничений $R_{\min}$, $W_{\max}$, $L_{\min}$, L_{dur} при каждом назначении рейса из объединённого расписания T. Изменения внесены и в пространство состояний: помимо признаков, аналогичных RL-MSA (признаки контрольных остановок cp, признаки текущего рейса l и признаки смен-кандидатов v), добавлены *смено-специфичные* признаки t^v_{end} (время до завершения смены) и t^l_{lunch} (время до обеда), необходимые для наблюдаемости трудовых ограничений. Дополнительно скорректирована процедура обучения агента: при использовании PPO для оценки преимущества применяется Generalized Advantage Estimation (GAE) [24], что повышает устойчивость обучения на длинных эпизодах последовательного назначения. В работе авторов [25] уже обсуждалось направление данного расширения. В настоящей статье оно уточняется за счёт конкретизации признаков состояния и практического сравнения различных подходов к реализации агента с использованием реальных данных.

3. Описание модели

В данном разделе представляется структура MDP-модели для решения задачи формирования графиков водительских смен с реальными ограничениями и критериями качества.

3.1. Пространство состояний

Состояние $s \in S$ на каждом шаге принятия решения формируется как конкатенация трёх групп признаков: признаков контрольных остановок, признаков текущего маршрута и признаков смен-кандидатов на рейс.

Признаки контрольных остановок. Для каждой контрольной остановки cp в транспортной сети формируется вектор признаков, характеризующий её текущее состояние:

- номер остановки n_{num}^{cp};
- число n_{lt}^{cp} предстоящих рейсов с отправлением с остановки cp;
- число n_{st}^{cp} предстоящих рейсов в следующие M минут (отражает спрос в ближайшее время; M является гиперпараметром);
- число n_a^{cp} смен, в рамках которых можно выполнить новый рейс в заданный момент отправления с остановки cp;
- число n_o^{cp} смен из числа n_a^{cp}, в рамках которых уже выполнен хотя бы один рейс.

Номер остановки n_{num}^{cp} нужен для установления связи с рейсом. Признаки n_{lt}^{cp} и n_{st}^{cp} используются для оценки спроса на рейсы для данной контрольной остановки в будущем. Признаки n_a^{cp} и n_o^{cp} используются для оценки числа доступных смен на контрольной остановке cp.

Признаки текущего маршрута. Для рейса, на который в данный момент назначается смена, формируются следующие признаки:

- номер начальной контрольной остановки d^l;
- номер конечной контрольной остановки a^l;
- время в пути h^l.

Данные признаки определяют контекст текущего решения и позволяют агенту учитывать специфику конкретного рейса при выборе смены.

Признаки смен-кандидатов. Для каждой смены-кандидата v формируется набор следующих признаков:

- статус использования смены, w_s^v: бинарный признак, где $w_s^v = 1$ означает, что смена уже выполнила хотя бы один рейс (используемая смена), иначе $w_s^v = 0$ (неиспользуемая смена);
- время отдыха, r^v: время, прошедшее с момента завершения последнего рейса. Если смена в данный момент выполняет рейс, то $r^v = 0$;
- текущая локация d^v: номер контрольной остановки, на которой в данный момент находится водитель, выполняющий смену v. Если водитель находится в рейсе, то $d^v = -1$;
- время холостой поездки, k^v: если водитель, выполняющий смену v, не находится на начальной контрольной остановке текущего рейса, необходима холостая поездка. Если холостая поездка не требуется, то $k^v = 0$;
- время, оставшееся до завершения смены, t_{end}^v;
- время, оставшееся до обеда, t_{lunch}^l.

Механизм маскирования действий. Для снижения размерности пространства состояний и ускорения обучения применяется механизм маскирования действий. Этот механизм формирует целевое множество смен V_{target} ограниченной мощности N_s, из которого агент выбирает смену для назначения на текущий рейс. Процесс отбора включает следующие этапы.

Классификация смен. Доступные смены разделяются на две категории:

- V_p — смены, находящиеся на начальной контрольной остановке текущего рейса и имеющие достаточное время отдыха ($r^v \geqslant R_{\min}$). Для данных смен холостая поездка не требуется;
- V_q — смены, находящиеся на других контрольных остановках ($k^v > 0$) и удовлетворяющие условию:

$$r^v > k^v + R_{\min}.$$

Для данных смен имеется достаточное время на отдых даже с учётом холостой поездки.

Внутренняя сортировка. Внутри каждой категории смены сортируются:

- используемые смены из V_p сортируются по убыванию времени отдыха r^v (смены с большим простоем имеют приоритет для повышения утилизации);

- используемые смены из V_q сортируются по возрастанию времени холостой поездки k^v (минимизация затрат на холостые поездки);
- неиспользуемые смены не сортируются, так как их выбор равнозначен.

Формирование целевого множества. Смены отбираются в следующем порядке приоритета до достижения мощности N_s:

- используемые смены из V_p (холостая поездка не требуется);
- используемые смены из V_q (требуется холостая поездка);
- неиспользуемые смены из V_p;
- неиспользуемые смены из V_q.

Итоговое состояние s формируется путём конкатенации векторов признаков всех контрольных остановок, признаков текущего маршрута и признаков всех смен из целевого множества V_{target}, после чего производится их сглаживание в единый вектор фиксированной размерности.

3.2. Пространство действий

Пространство действий A определяет множество решений, доступных агенту на каждом шаге. Действие $a \in A$ представляет собой выбор конкретной смены из целевого множества V_{target} для назначения на текущий рейс.

Формально, $A = \{a_1, a_2, \ldots, a_{N_s}\}$, где каждое действие a_i соответствует назначению i-й смены из V_{target} на рейс. Размерность пространства действий фиксирована и равна N_s — мощности целевого множества смен-кандидатов (является гиперпараметром).

3.3. Функция награды

Пошаговая награда. На каждом шаге t агент получает пошаговую награду r_t, состоящую из нескольких компонент. Штраф за использование новой смены:

$$r_t^{new} = \begin{cases} 1, & \text{если на рейс назначена активная смена,} \\ 0, & \text{если на рейс назначена новая смена.} \end{cases}$$

Этот компонент стимулирует агента максимально использовать уже задействованные смены, минимизируя общее количество смен. Штраф за холостую поездку:

$$r_t^{deadhead} = k^v,$$

где k^v — время холостой поездки выбранной смены v.

Поощрение использования смен с большим временем отдыха. Чтобы повысить утилизацию смен, агенту предлагается выбирать смены с большим временем отдыха после последнего рейса r^v. Для этого задаётся вознаграждение r_k. Если выбранная смена не использовалась ранее, r_k полагается равным 0. Иначе подсчитывается количество N_o активных смен во множествах V_p и V_q. Далее смены сортируются по убыванию в соответствии со временем отдыха r^v. Если ранг выбранной смены v равен p^v, тогда r_k определяется следующим образом:

$$r_t^{idle} = \frac{N_o - p^v}{N_o}.$$

Чем меньше p^v, тем больше время отдыха смены v и тем быстрее её нужно выбрать, поэтому вознаграждение r_k, получаемое при выборе этой смены, больше.

Штраф за перегон смены с загруженной остановки. Когда выбирается смена из множества V_q, чтобы избежать выполнения холостого рейса из контрольной остановки с высоким будущим спросом

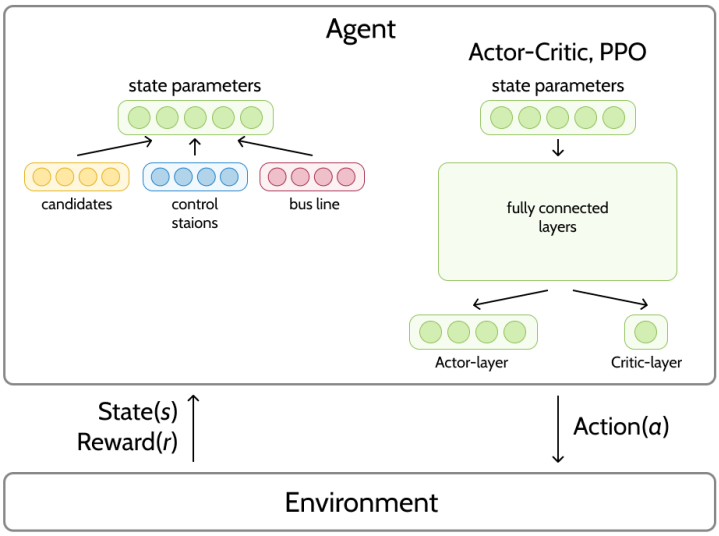

Fig. 1. The RL agent architecture

Рис. 1. Архитектура RL-агента

в контрольную остановку с низким будущим спросом, вычисляется степень спроса U_{cp} для контрольной остановки, где водитель на смене сейчас находится U_1, и конечной контрольной остановки линии маршрута, соответствующего текущему времени отправления U_2. Для контрольной остановки cp величина U_{cp} задаётся следующим образом:

$$U_{cp} = \frac{n_{st}^{cp}}{n_o^{cp} + 1}.$$

Чем больше n_{st}^{cp}, тем выше будущий спрос на рейсы на контрольной остановке cp. Чем меньше n_o^{cp}, тем меньше водителей находится на данной контрольной остановке. Наконец, чем больше U_{cp}, тем больше водителй требуется контрольной остановке cp. Если U_{cp} больше, чем U_{cp+1}, накладывается штраф r_t^{util} ($r_t^{util} = 1$). Когда выбирается смена из множества V_p, штраф равен 0. Итоговая пошаговая награда:

$$r_t = -w_1 \cdot r_t^{new} - w_2 \cdot r_t^{deadhead} + w_3 \cdot r_t^{idle} - w_4 \cdot r_t^{util},$$

где w_1, w_2, w_3, w_4 — весовые коэффициенты, определяющие относительную важность каждого компонента целевой функции.

Финальная награда. По завершении эпизода (после назначения всех рейсов из расписания T) агент получает финальную награду r_{final}:

$$r_{final} = -u_1 \cdot N_{shifts} - u_2 \cdot T_{deadhead}, \tag{1}$$

где u_1, u_2 — весовые коэффициенты, определяющие относительную важность каждого компонента целевой функции.

3.4. Архитектура агента

Агент реализован на основе архитектуры Actor-Critic с использованием алгоритма PPO (рис. 1). Также в рамках сравнительного анализа использовались три варианта RL-агентов: Actor-Critic (PPO), REINFORCE и DQN. Для Actor-Critic (PPO) скрытое представление State-Net подавалось на два выходных слоя: Actor, формирующий распределение вероятностей по действиям размерности N_s, и Critic, оценивающий ценность состояния. Для REINFORCE использовался только Actor-слой. Для DQN

Table 1. Route information **Таблица 1.** Информация о маршрутах

Маршрут	Длит. рейса, мин	Число рейсов
8: ТРК Вернисаж — 15-й микрорайон	79	200
11: 15-й микрорайон — Яр.-Главный	49	176
32: 15-й микрорайон — Гипермаркет Глобус	50	192
53: ТРК Вернисаж — Гипермаркет Глобус	48	135
76: ТРК Вернисаж — Яр.-Главный	35	133
99: Гипермаркет Глобус — Яр.-Главный	50	138

скрытое представление State-Net подавалось на выходной слой, возвращающий вектор оценок $Q(s, a)$ для всех действий из множества кандидатов.

Для уменьшения сложности модели Actor и Critic используют общую сеть State-Net, представляющую собой последовательность полносвязных слоёв с ReLU-активацией ($128 \to 64 \to 32$ нейрона). На вход State-Net подаётся вектор признаков состояния, а на выходе получается скрытое представление, которое затем подаётся на два независимых выходных слоя. Actor-слой ($32 \to N_s$) выдаёт вероятностное распределение по действиям, определяя политику агента. Critic-слой ($32 \to 1$) выдаёт скаляр, представляющий оценку ценности текущего состояния.

Выбор подхода с использованием State-Net обусловлен тем, что данная конфигурация была описана в работе [23] и показала устойчивую обучаемость. В настоящей работе архитектура сохранена в качестве бэйзлайна, чтобы обеспечить воспроизводимость и минимизировать риск переобучения при ограниченном объёме доступных экземпляров расписания, поскольку увеличение числа параметров сети в таких условиях может ухудшать обобщающую способность политики.

3.5. Программная реализация

Система реализована на языке Python с использованием библиотеки PyTorch для построения и обучения нейронных сетей, NumPy для векторных операций и Pandas для обработки табличных данных расписания.

В рамках исследования была разработана симуляционная среда, реализующая описанную MDP-модель. Среда моделирует последовательное назначение смен на рейсы с учётом всех ограничений: рабочего времени, обеденных перерывов и минимального отдыха между рейсами. В начальный момент времени $t = 0$ фиксируется первый рейс из расписания. Агент, наблюдая текущее состояние, выбирает смену из целевого множества V_{target}. Среда реагирует, выдавая агенту награду и переходя к следующему рейсу. Этот процесс продолжается до распределения всех рейсов по водительским сменам, после чего агент получает финальную награду и эпизод завершается.

4. Экспериментальное исследование

4.1. Описание данных

Эксперименты проводились на реальных данных транспортной сети города Ярославль. Данные включают информацию о 4 контрольных остановках, 6 маршрутах с 12 маршрутными линиями и расписание движения, содержащее в сумме 974 рейса (таблица 1).

Матрица времени перегонов между остановками представлена в таблице 2.

Параметры смен заданы следующим образом:

- рабочее время $W_{\max}$ — 8 часов;
- обед $L_{\max}$ — 1 час;
- время до обеда $L_{\min}$ — 4 часа;
- минимальный отдых между рейсами $R_{\min}$ — 5 минут.

Table 2. The deadhead time matrix between stops, min

Таблица 2. Матрица перегонов между остановками, мин

Из / В	15-й мкр	Яр.-Главный	Глобус	Вернисаж
15-й микрорайон	0	21	29	28
Яр.-Главный	22	0	25	23
Глобус	31	25	0	34
Вернисаж	30	24	35	0

Table 3. Reward function weight coefficients

Таблица 3. Весовые коэффициенты функции награды

u_1	u_2	w_1	w_2	w_3	w_4
4.0	0.1	4.0	0.1	2.5	1.0

Table 4. PPO agent training hyperparameters

Таблица 4. Гиперпараметры обучения PPO-агента

Параметр	**Значение**
Число эпизодов	500
Learning rate	3×10^{-4}
Discount factor γ	0.99
GAE λ	0.95
Clip ϵ	0.2
Entropy coefficient	0.01
Value coefficient	0.5
Mini-batch size	64
Epochs per update	4

4.2. Настройка гиперпараметров

Весовые коэффициенты функции награды задавались в два этапа. Коэффициенты финальной награды u_1 и u_2 были взяты из работы [23], поскольку данный член награды непосредственно соответствует прикладным целям минимизации числа смен и суммарного времени холостых поездок. Коэффициенты пошаговой награды w_1, w_2, w_3, w_4 уточнялись с использованием исчерпывающего поиска по гиперпараметрам (grid search).

Для этого задавалась дискретная решётка значений: $w_1 \in \{1, 2, 4, 6, 8\}$, $w_2 \in \{0.01, 0.05, 0.1, 0.2, 0.5\}$, $w_3 \in \{1, 1.5, 2, 2.5, 3, 4.5, 4\}$, $w_4 \in \{0.5, 1, 1.5, 2, 2.5, 3\}$, после чего перебирались все комбинации из данной решётки. Для каждой комбинации выполнялось обучение агента, и далее проводилась оценка на 10 независимых тестовых запусках. В качестве критерия отбора использовалось среднее значение r_{final}. Итоговые значения коэффициентов приведены в таблице 3.

Размер множества кандидатов $N_s = 8$. Каждый эпизод состоит из 974 шагов, соответствующих последовательному назначению смен на все рейсы расписания. Гиперпараметры обучения PPO-агента в таблице 4 выбирались как типичные значения, рекомендуемые в оригинальной работе [16] и широко используемые в практических реализациях.

4.3. Сравнительный анализ результатов обучения

На рис. 2 представлена динамика обучения PPO-агента. Стабильная политика достигается приблизительно к 100-му эпизоду. Также проведено сравнение с альтернативными алгоритмами: DQN, REINFORCE, жадным алгоритмом и генетическим алгоритмом (GA). На рис. 3 показаны кривые обучения для REINFORCE и DQN.

Результаты сравнительного анализа представлены в таблице 5. $J(\pi)$ обозначает суммарную награду, набранную обученным агентом за эпизод. В рамках эксперимента значение $J(\pi)$ оценивалось

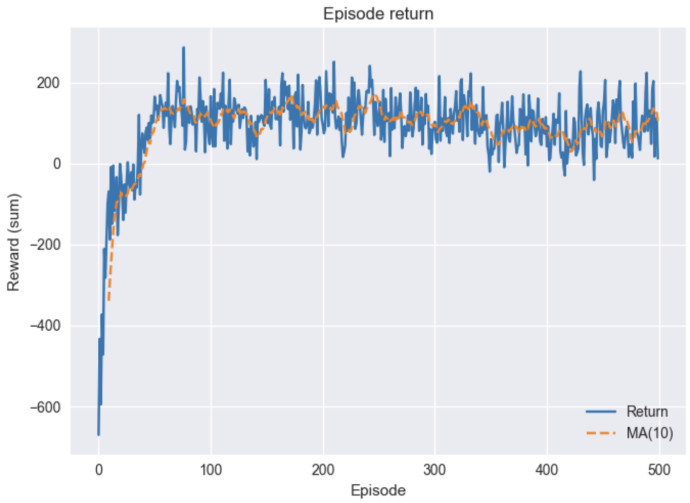

Fig. 2. Reward dynamics by episode (PPO)

Рис. 2. Динамика награды по эпизодам (PPO)

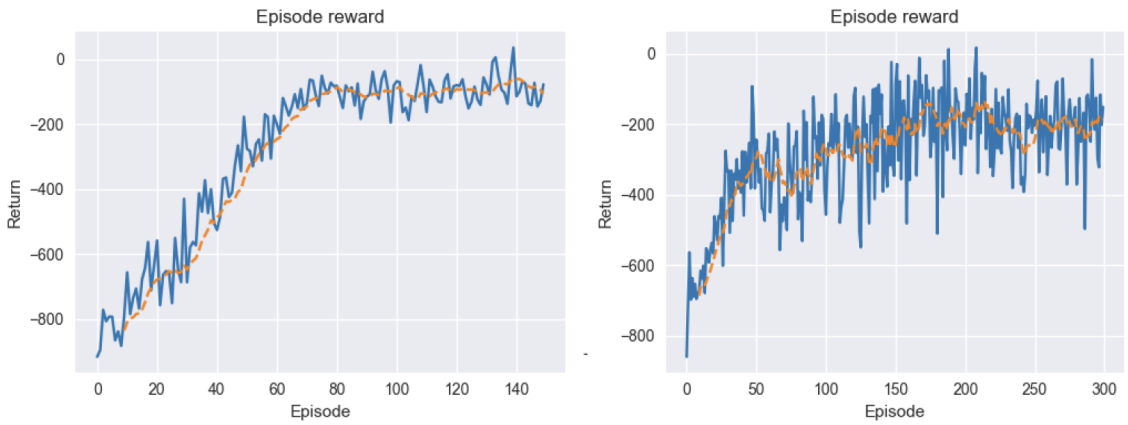

Fig. 3. Training dynamics: REINFORCE (left) and DQN (right)

Рис. 3. Динамика обучения: REINFORCE (слева) и DQN (справа)

Table 5. Algorithm comparison

Таблица 5. Сравнение алгоритмов

Алгоритм	$J(\pi)$	$\lvert r_{final}\rvert$	N_{shifts}	$T_{deadhead}$, **мин**
Actor-Critic (PPO)	**240.68**	**775.5**	168	1031
REINFORCE	101.90	811.1	168	1362
DQN	−297.64	885.2	**167**	2051
Генетический алгоритм	—	809.4	180	**857**
Жадный алгоритм	—	1069.7	172	3817

как среднее по 10 независимым запускам агента. Величина $|r_{final}|$ (1) представляет модуль финальной награды и интерпретируется как оценка операционных расходов. В таблице приведено среднее значение $|r_{final}|$. Показатель N_{shifts} равен числу задействованных смен в сформированных графиках смен, а $T_{deadhead}$ — суммарному времени холостых поездок. Поскольку N_{shifts} и $T_{deadhead}$ характеризуют конкретные сформированные графики смен, в таблице они приведены для наилучших графиков смен, которые смог получить агент.

PPO демонстрирует наилучшие результаты по совокупности метрик. Несмотря на то что DQN задействовал минимальное число смен (167), время холостых поездок оказалось значительно выше

(2051 мин против 1031 мин у PPO), что нивелирует данное преимущество и приводит к худшему значению целевой функции.

Превосходство PPO объясняется несколькими факторами. Во-первых, механизм ограничения изменений политики (gradient clipping) в PPO предотвращает резкие изменения политики, что критично для задач с разреженными наградами и длинными эпизодами. Одной из главных проблем методов policy gradient является чувствительность к размеру шага обучения: слишком маленький шаг снижает эффективность обучения, увеличивая вероятность застревания в локальном оптимуме, а слишком большой шаг может привести к коллапсу политики. PPO решает эту проблему, вводя ограничение на то, насколько сильно новая политика может отличаться от старой за один шаг обновления. Во-вторых, метод GAE, используемый в PPO, обеспечивает оптимальный баланс между смещением и дисперсией оценки преимущества, что ускоряет обучение. В-третьих, в отличие от DQN, PPO естественным образом поддерживает стохастические политики, что способствует исследованию пространства решений и помогает избегать проблем, когда похожие состояния требуют разных действий.

DQN показал нестабильное обучение с высокой дисперсией награды между эпизодами. Это связано с тем, что алгоритм оптимизирует Q-функцию, а не политику напрямую, что приводит к накоплению ошибок аппроксимации в длинных эпизодах.

REINFORCE продемонстрировал промежуточные результаты, однако высокая дисперсия оценки градиента замедляет сходимость и приводит к субоптимальным решениям по критерию времени холостых поездок.

Что касается жадного алгоритма, который на каждом шаге выбирал смену с наибольшим временем отдыха, то его ключевое отличие от RL-агента состоит в том, что он принимает решение исключительно на основе текущего состояния, не учитывая будущие последствия выбора. Это приводит к неоптимальному распределению смен по остановкам: алгоритм может перенаправить смену с остановки с высоким будущим спросом на остановку с низким, что впоследствии потребует создания дополнительных смен или увеличит время холостых поездок. Единственное преимущество жадного алгоритма перед PPO состоит в скорости вычислений, так как он не требует обучения модели и может быть реализован на основе простых правил.

Для дополнительной оценки качества RL-подхода был реализован генетический алгоритм как более сильный метаэвристический бейзлайн. В данной реализации хромосома представляет собой вектор вещественных приоритетов (по одному на каждый рейс расписания), определяющих выбор смены из отсортированного множества допустимых кандидатов при последовательном построении графиков смен. Эволюционный поиск проводился с популяцией в 50 особей на 100 поколениях. Было выполнено 10 независимых запусков и получено среднее значение $|r_{final}| = 809.4$. Лучшее решение использует 180 смен при суммарном времени холостых поездок 857 минут. На рис. 4 представлена динамика сходимости генетического алгоритма на одном из запусков.

Генетический алгоритм достиг среднего значения $|r_{final}| = 809.4$, что довольно близко к среднему результату PPO (775.5). Однако структура полученного решения отличается: лучшее решение генетического алгоритма использует 180 смен при суммарном времени холостых поездок 857 минут, тогда как лучшее решение PPO формирует 168 смен с 1031 минутой холостых поездок. Таким образом, генетический алгоритм минимизирует перемещения между контрольными остановками ценой большего числа задействованных смен, в то время как PPO находит более сбалансированное решение с существенно меньшим числом смен. Учитывая, что в целевой функции (1) вклад каждой дополнительной смены значительно превышает вклад минуты холостой поездки, результат PPO предпочтительнее с точки зрения операционной эффективности: меньшее число смен означает меньшую потребность в водительском и транспортном составе.

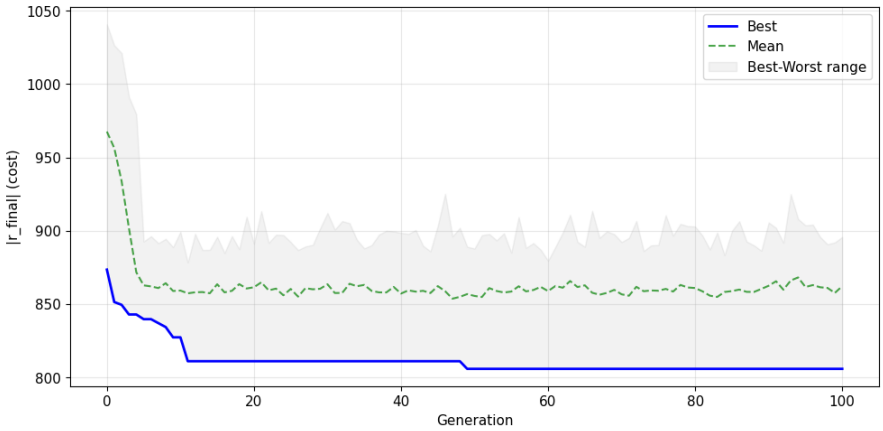

Fig. 4. GA convergence

Рис. 4. Сходимость генетического алгоритма

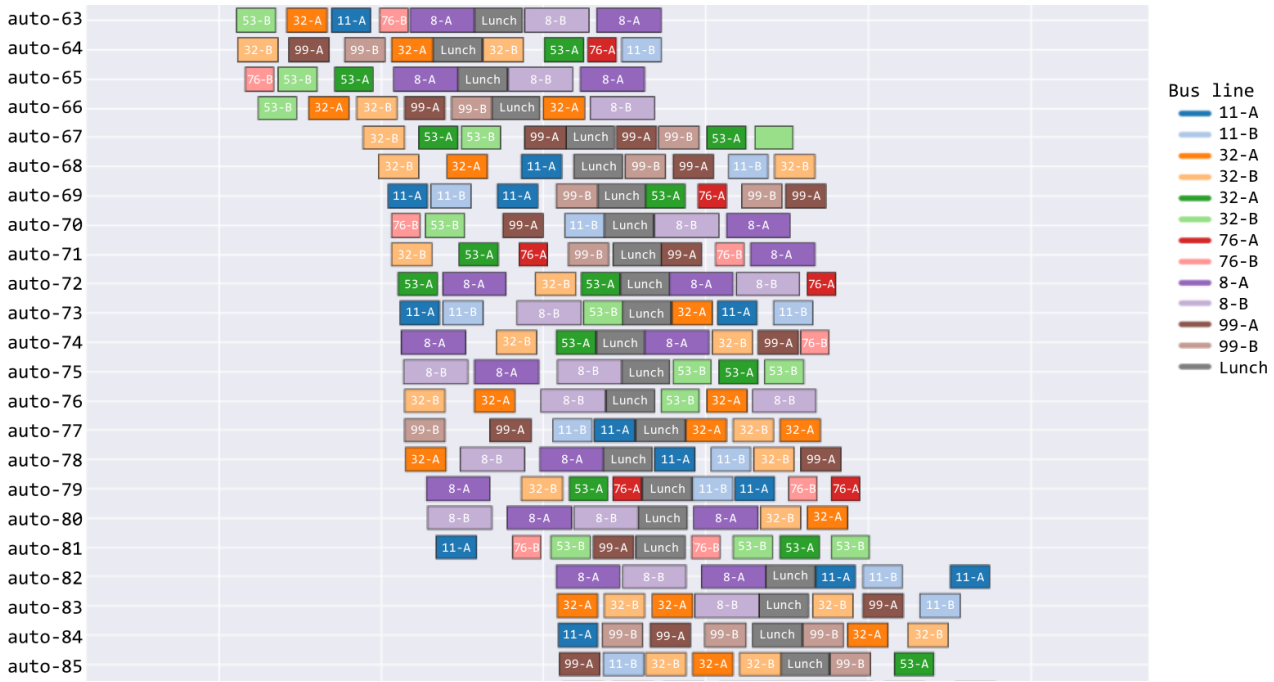

Fig. 5. Fragment of Gantt chart with shift schedules

Рис. 5. Фрагмент диаграммы Ганта с графиками смен

На рис. 5 представлен фрагмент диаграммы Ганта с полученными графиками смен агентом, основанным на архитектуре Actor-Critic с алгоритмом PPO. Буквами A и B обозначены разные направления по маршрутам. Схожими цветами обозначены линии одного маршрута. Серым блоком обозначены обеденные перерывы.

5. Возможности использования и внедрения

Данный подход в перспективе может применяться в следующих областях.

Городской пассажирский транспорт. Автоматизация планирования работы водителей автобусов, троллейбусов, трамваев с учётом специфических требований конкретного предприятия. Система позволяет оперативно перестраивать графики при изменении расписания движения, сезонных колебаниях пассажиропотока или кадровых изменениях.

Железнодорожный транспорт. Адаптация подхода для составления графиков работы машинистов и проводников с учётом междугородных маршрутов, требований к квалификации персонала и специфики ночных смен.

Логистические компании. Оптимизация работы водителей грузового транспорта с учётом тахографических ограничений и требований к соблюдению режима труда и отдыха.

По сравнению с традиционными методами оптимизации предлагаемый подход обладает рядом существенных преимуществ. Обученный агент способен генерировать решения для новых экземпляров задачи без повторного обучения или с небольшим дообучением, что критично при частых изменениях расписания движения, сезонных колебаниях пассажиропотока или кадровых изменениях. Время генерации решения линейно зависит от числа рейсов, в то время как методы точной оптимизации часто имеют экспоненциальную сложность, делая их неприменимыми для крупномасштабных задач. Добавление новых ограничений, связанных со спецификой конкретного транспортного предприятия, реализуется через модификацию функции награды без изменения архитектуры системы. Агент может быть дообучен на новых данных при изменении характеристик транспортной сети, что обеспечивает долгосрочную актуальность решения. Наконец, система может быть встроена в существующие автоматизированные системы управления транспортом в качестве модуля поддержки принятия решений.

Перспективными направлениями развития данного подхода являются решение интегрированной задачи совместной оптимизации расписания движения и графиков смен, что позволит достичь глобально оптимальных решений вместо последовательного решения двух связанных подзадач. Актуальным является разработка политик, устойчивых к неопределённостям реальной эксплуатации: задержкам рейсов, поломкам транспортных средств и неявкам водителей. Для крупных транспортных сетей перспективно применение многоагентного обучения с подкреплением, позволяющего декомпозировать задачу на независимо управляемые подсистемы. Наконец, исследование трансферного обучения позволит переносить знания между различными транспортными сетями для ускорения обучения агентов при развёртывании системы в новых условиях.

Заключение

В работе представлен подход к решению задачи автоматизированного составления графиков водительских смен с использованием методов глубокого обучения с подкреплением. Описана MDP-модель, учитывающая специфические ограничения транспортной отрасли: рабочее время, обеденные перерывы, минимальный отдых между рейсами.

Предложена архитектура агента на основе метода Actor-Critic с алгоритмом PPO, демонстрирующая стабильное обучение и высокое качество получаемых решений. Экспериментальное исследование на реальных данных транспортной сети города Ярославль показало, что PPO превосходит альтернативные алгоритмы (DQN, REINFORCE).

Агент, обученный методом PPO, достиг средней награды 240.68, обеспечив в лучшем составленном графике смен покрытие расписания из 974 рейсов с использованием 168 смен и суммарным временем холостых поездок в 1031 минуту. По сравнению с жадным алгоритмом достигнуто сокращение числа смен на 2.3 % и времени холостых поездок в 3.7 раза. Дополнительное сравнение с генетическим алгоритмом показало, что PPO достигает лучшего значения целевой функции, формируя решение с числом задействованных смен, меньшим на 7.1 %, что предпочтительнее с точки зрения операционной эффективности транспортного предприятия. Результаты подтверждают возможность применения методов глубокого обучения с подкреплением для решения задач транспортной оптимизации.

References

[1] A. Wren, "Scheduling, timetabling and rostering — a special relationship?", in *Practice and Theory of Automated Timetabling*, ser. Lecture Notes in Computer Science, vol. 1153, 1996, pp. 46–75. DOI: 10.1007/3-540-61794-9_51.

[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd edition. Cambridge, MA: MIT Press, 2018.

[3] R. E. Marsten and F. Shepardson, "Exact solution of crew scheduling problems using the set partitioning model", *Networks*, vol. 11, no. 2, pp. 165–177, 1981. DOI: 10.1002/net.3230110208.

[4] M. Desrochers and F. Soumis, "A column generation approach to the urban transit crew scheduling problem", *Transportation Science*, vol. 23, no. 1, pp. 1–13, 1989. DOI: 10.1287/trsc.23.1.1.

[5] C. C. Ribeiro and F. Soumis, "A column generation approach to the multiple-depot vehicle scheduling problem", *Operations Research*, vol. 42, no. 1, pp. 41–52, 1994. DOI: 10.1287/opre.42.1.41.

[6] T. A. Feo and M. G. C. Resende, "Greedy randomized adaptive search procedures", *Journal of Global Optimization*, vol. 6, pp. 109–133, 1995. DOI: 10.1007/BF01096763.

[7] T. Park and K. R. Ryu, "Crew pairing optimization by a genetic algorithm with unexpressed genes", *Journal of Intelligent Manufacturing*, vol. 17, no. 4, pp. 375–383, 2006. DOI: 10.1007/s10845-005-0011-z.

[8] F. Glover, "Future paths for integer programming and links to artificial intelligence", *Computers & Operations Research*, vol. 13, no. 5, pp. 533–549, 1986. DOI: 10.1016/0305-0548(86)90048-1.

[9] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing", *Science*, vol. 220, no. 4598, pp. 671–680, 1983. DOI: 10.1126/science.220.4598.671.

[10] J. Ma, C. Song, A. Ceder, T. Liu, and W. Guan, "Fairness in optimizing bus-crew scheduling process", *PLoS One*, vol. 12, no. 11, e0187623, 2017. DOI: 10.1371/journal.pone.0187623.

[11] R. S. K. Kwan, A. S. K. Kwan, and A. Wren, "Evolutionary driver scheduling with relief chains", *Evolutionary Computation*, vol. 9, no. 4, pp. 445–460, 2001. DOI: 10.1162/10636560152642869.

[12] Q. Cappart, T. Che, Z. Kulkarni, M. Morabit, and L.-M. Rousseau, "Combining reinforcement learning and constraint programming for combinatorial optimization", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 3677–3687. DOI: 10.1609/aaai.v35i5.16484.

[13] V. Mnih *et al.*, "Human-level control through deep reinforcement learning", *Nature*, vol. 518, no. 7540, pp. 529–533, 2015. DOI: 10.1038/nature14236.

[14] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning", *Machine Learning*, vol. 8, no. 3, pp. 229–256, 1992. DOI: 10.1007/BF00992696.

[15] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning", in *Proceedings of the 33rd International Conference on Machine Learning*, PMLR, 2016, pp. 1928–1937.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, *Proximal policy optimization algorithms*, 2017. DOI: 10.48550/arXiv.1707.06347. arXiv: 1707.06347 [cs.LG].

[17] W. Kool, H. van Hoof, and M. Welling, "Attention, learn to solve routing problems!", in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019, pp. 1–25.

[18] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks", in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 28, 2015, pp. 2692–2700.

[19] L. Kenworthy, S. Nayak, C. Chin, and H. Balakrishnan, "NICE: Robust scheduling through reinforcement learning-guided integer programming", *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 9, 2022. DOI: 10.1609/aaai.v36i9.21218.

[20] M. Nagayoshi and H. Tamaki, "Constructive nurse scheduling using reinforcement learning considering variations in nurse work patterns", in *Proceedings of the International Conference on Artificial Life and Robotics (ICAROB 2024)*, 2024, pp. 325–328.

[21] X. Zhang *et al.*, "Multi-agent deep Q-network-based metaheuristic algorithm for nurse rostering problem", vol. 87, 2024, p. 101 547. DOI: 10.1016/j.swevo.2024.101547.

[22] X. Ding, H. Liu, and X. Chen, "Deep reinforcement learning for solving multi-period routing problem with binary driver-customer familiarity", in *Proceedings of the 2023 5th International Conference on Internet of Things, Automation and Artificial Intelligence*, Association for Computing Machinery, 2024, pp. 125–129. DOI: 10.1145/3653081.3653103.

[23] Y. Liu and X. Zuo, *RL-MSA: A reinforcement learning-based multi-line bus scheduling approach*, 2024. DOI: 10.48550/arXiv.2403.06466. arXiv: 2403.06466 [cs.LG].

[24] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, *High-dimensional continuous control using generalized advantage estimation*, 2018. DOI: 10.48550/arXiv.1506.02438. arXiv: 1506.02438 [cs.LG].

[25] S. V. Goncharov and I. S. Vojteshenko, "K voprosu naznacheniya voditeley na reysy gorodskogo obshchestvennogo transporta na osnove obucheniya s podkrepleniem", in *Informatsionnye tekhnologii i sistemy 2025 (ITS 2025)*, in Russian, pp. 45–46.