УДК 519.85:656.7

DOI: http://doi.org/10.25728/pu.2023.2.1

МЕТОДЫ РЕШЕНИЯ ЗАДАЧ ПЛАНИРОВАНИЯ И РЕГУЛИРОВАНИЯ ПОТОКОВ ВОЗДУШНОГО ДВИЖЕНИЯ

Ч. 2. Применение методов глубокого обучения с подкреплением

Е.Л. Кулида, В.Г. Лебедев

Аннотация. Рассмотрены задачи повышения безопасности и эффективности организации воздушного движения. Главной проблемой при решении задач обнаружения и разрешения конфликтов традиционными методами оптимизации является время вычислений – требуются десятки и даже сотни секунд, но в реальных ситуациях времени на реакцию не так много. В последнее время широкое распространение получило применение глубокого обучения с подкреплением, поскольку с его помощью удается за приемлемое время решать задачи принятия решений с нелинейностью и большой размерности. В последние несколько лет появились исследовательские работы по использованию глубокого обучения с подкреплением для решения задач в области управления воздушным движением. В обзоре уделяется особое внимание применению этого перспективного подхода для решения следующих задач: обнаружения и разрешения конфликтов между воздушными судами, крупномасштабной задачи снижения сложности воздушного движения в воздушном пространстве страны или континента, повышения эффективности использования взлетно-посадочных полос аэропортов на основе улучшенного планирования посадок воздушных судов.

Ключевые слова: управление воздушным движением, стратегическое планирование четырехмерных траекторий, обнаружение и разрешение конфликтов воздушных судов, обучение с подкреплением.

ВВЕДЕНИЕ

В связи с ростом интенсивности воздушного движения и перегруженностью крупнейших аэропортов растет запрос на автоматизацию работы авиадиспетчеров путем разработки систем поддержки принятия решений и автоматизированных систем управления воздушным движением. В первой части обзора [1] была рассмотрена задача минимизации числа потенциальных конфликтов между воздушными судами (ВС).

В работе [2] представлен обзор и современные тенденции применения искусственного интеллекта в управлении воздушным движением, составленный на основе материалов конференций и публикаций в журналах с высоким рейтингом, представляющих предметную область. Несмотря на значительный прогресс исследовательских работ в области искусственного интеллекта для управления воздушным движением, он пока не стал «полностью работоспособным» для конечных пользователей. Медленный прогресс в применении искусственного интеллекта в области управления воздушным движением объясняется тем фактом, что рассматриваемая область является критической, здесь на карту поставлена жизнь, и безопасность является главным приоритетом. В настоящее время безопасность в управлении воздушным движением достигается при помощи участия человека в контуре управления и, скорее всего, как утверждают авторы, будет развиваться путем проектирования систем, ориентированных на человека, требующих, чтобы эти системы были понятны конечному пользователю и адаптировались к его психологическому состоянию. Для этого необходимо перейти к более ориентированному на пользователя объяснимому искусственному интеллекту, где система искусственного интеллекта и конечный пользователь смогут понимать друг друга и взаимодействовать друг с другом.

Подходы, основанные на оптимизации, часто являются дорогостоящими с точки зрения вычис-

лений, что ограничивает их применение. Впечатляющие результаты получены в нескольких исследовательских работах, связанных с управлением воздушным движением на основе глубокого обучения с подкреплением [3].

В докладе [4] была впервые сформулирована модель на основе обучения с подкреплением и представлен агент искусственного интеллекта для смягчения конфликтов и минимизации задержек ВС при достижении контрольных точек. В работах [5, 6] исследуется влияние различных уровней неопределенности окружающей среды и плотности трафика на производительность модели на основе обучения с подкреплением для разрешения конфликтов между ВС.

Если решения, предлагаемые при автоматическом разрешении конфликтов, не соответствуют мышлению или предпочтениям диспетчеров, они вряд ли будут приняты. В статье [7] разработан интерактивный агент искусственного интеллекта на основе обучения с подкреплением, которому предоставлялись маневры разрешения конфликтов, используемые диспетчером-человеком. Такой подход может, по мнению авторов, повысить уровень доверия диспетчера предлагаемым агентом решениям. В докладе [8] предложен гибридный алгоритм, который использует известные геометрические методы на этапе глубокого обучения с подкреплением для разрешения конфликтов в воздушном пространстве на малой высоте.

Эти подходы эффективны в условиях низкой плотности воздушного движения, но централизованные архитектуры не справляются с высокой плотностью воздушного движения при увеличении числа конфликтующих самолетов. В большинстве сложных систем считается, что распределенное принятие решений более эффективно, чем централизованное управление. Важнейшей задачей для обеспечения распределенного принятия решений при управлении воздушным движением является разработка системы, предоставляющей ВС рекомендации для обеспечения безопасного разделения и устранения неопределенности в режиме реального времени. Было предложено несколько многоагентных подходов для работы в условиях высокой плотности воздушного движения. В работах [9–11] показано, как децентрализованная система позволяет многим агентам иметь доступ ко всей информации о ВС в секторе с помощью масштабируемого и эффективного способа для достижения высокой пропускной способности в условиях неопределенности. Для обучения агентов используется одна нейронная сеть и централизованное обучение, а также децентрализованная схема принятия решений. Многие из предлагаемых агентов на основе обучения с подкреплением должны обучаться в среде с фиксированным количеством конфликтующих ВС. По мере увеличения числа конфликтующих ВС вычислительная сложность обучения быстро возрастает. В докладе [12] для разрешения конфликтов между ВС предлагается метод глубокого обучения с подкреплением на основе изображений. Глубокое обучение на основе изображений в значительной степени решает проблему масштабируемости. Алгоритм может обрабатывать произвольное количество ВС, поскольку используются изображения, а не состояния ВС. В работе [13] представлена модель автономного управления воздушным движением в свободном воздушном пространстве, предотвращающая столкновения между ВС. Предложен графический нейросетевой подход к разрешению конфликтов в свободном воздушном пространстве путем представления каждого ВС в виде узла на графе. Этот подход также позволяет справляться с произвольным количеством ВС.

Предполагается, что глубокое обучение с подкреплением будет играть существенную роль в построении будущих систем управления воздушным движением, однако необходимы дополнительные исследования. При использовании глубокого обучения с подкреплением в реальных приложениях наиболее существенны две проблемы.

Первая проблема – это безопасность систем управления воздушным движением. Современные модели используют глубокие нейронные сети в качестве аппроксиматоров функций. Было обнаружено, что глубокие нейронные сети уязвимы для состязательных примеров [14, 15] – тщательно продуманных квазинезаметных возмущений, которые вводят глубокую нейронную сеть в заблуждение при добавлении этих возмущений к ее входным данным. Более того, состязательные примеры также появляются в реальном мире без участия какого-либо злоумышленника или злонамеренно выбранного шума [16]. Следовательно, необходимо тщательно изучить механизмы состязательной атаки, способы обнаружения ошибок или нежелательного поведения алгоритмов искусственного интеллекта и методы их преодоления.

Вторая проблема – это объяснимость. Процесс принятия решений по технологии глубокого обучения с подкреплением непрозрачен, и отсутствие прозрачности делает невозможным для пилотов и диспетчеров понимание их внутреннего режима работы. Природа «черного ящика» модели может помешать пользователям поверить в прогнозируемые результаты, особенно когда модель используется для принятия ключевых решений.

В работе [17] представлена модель под названием safety-aware deep *Q* networks (учитывающая безопасность сеть глубокого обучения), в которой две отдельные нейронные сети совместно исследуют безопасность и затраты на оптимизацию. Авторы утверждают, что их работа является первой, в которой предусматривается уязвимость модели от состязательных атак, и что модель безопасна и хорошо объяснима.

Узким местом в системе управления воздушным движением является пропускная способность взлетно-посадочных полос крупных аэропортов. Операции управления маневрированием ВС в зоне аэропортов, такие как контроль прибытия, планирование последовательности и времени посадок, выполняются авиадиспетчерами. В работе [18] представлен системный обзор прошлых и самых современных теоретических исследований, связанных с проблемой посадки ВС в аэропортах, имеющих одну или несколько взлетно-посадочных полос, и их сравнительное исследование.

Далее в обзоре более детально рассматривается подход на основе глубокого обучения с подкреплением для решения следующих задач: обнаружения и разрешения конфликтов между ВС, крупномасштабной задачи снижения сложности воздушного движения в воздушном пространстве страны или континента, повышения эффективности использования взлетно-посадочных полос аэропортов на основе улучшенного планирования посадок ВС.

1. ОБНАРУЖЕНИЕ И РАЗРЕШЕНИЕ КОНФЛИКТОВ НА ОСНОВЕ ГЛУБОКОГО ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ

1.1. Подход к решению задач на основе глубокого обучения с подкреплением

Обучение с подкреплением [19] – это метод, при котором агент взаимодействует с окружающей средой с целью максимизации долгосрочного вознаграждения. Его можно рассматривать как марковский процесс принятия решений (S, A, T, R, γ), где S – множество состояний среды; A – множество действий агента; T – функция, определяющая вероятность перехода из одного состояния в другое; R – функция вознаграждения; γ – коэффициент дисконтирования.

В момент обучения t агент находится в состоянии $S_t \in S$ и генерирует действие $A_t \in A$, следуя политике π: $S \times A \to R$. Затем агент получает вознаграждение R_t и переходит в следующее состояние S_{t+1}. Цель агента заключается в том, чтобы изучить политику $\pi: S \to A$, которая определяет, какое действие нужно использовать в каждом состоянии, чтобы максимизировать суммарное вознаграждение.

Состояния оцениваются на основе функции $V(S)$. Функция оценки состояния S_t обновляется по формуле

$$V(S_t) \leftarrow V(S_t) + \alpha\left(R_t + \gamma V(S_{t+1}) - V(S_t)\right),$$

где α – скорость обучения.

Алгоритм строится на основе двух нейронных сетей: сети критика и сети исполнителя. Сеть критика используется для обновления параметров функции значений w, сеть исполнителя используется для обновления параметров политики θ [20].

Поскольку пространство состояний бесконечно, используется аппроксимация функции оценки состояний. Для аппроксимации применяется глубокое обучение нейронной сети [21]:

$$\hat{V}(S, w) \approx V_n(S),$$

где w – веса нейронов. Формула обновления одношагового метода с временны́ми различиями с использованием приближенной функции определяется так:

$$w \leftarrow w + \alpha\left(R_t + \gamma\hat{V}(S_{t+1}, w) - \hat{V}(S_t, w)\right)\nabla_w\hat{V}(S_t, w).$$

Для выбора действия используется политика $\pi(a_s, \theta)$. Для пространства непрерывных действий обычно применяется алгоритм градиента политики [22] на основе градиентного спуска. Дифференцируемая политика определяется как $\pi(a_s, \theta)$, и на каждом временно́м шаге параметры θ обновляются следующим образом:

$$\theta \leftarrow \theta + \alpha\nabla_\theta \ln\pi(A_t|S_t, \theta)V(S_t).$$

1.2. Описание задачи обнаружения и разрешения конфликтов [23]

Рассматриваются N ВС в конкретном сценарии воздушного движения, показанном на рис. 1. Из них $N-1$ находятся в секторе с радиусом L, дополнительный самолет влетает в сектор. Каждый самолет имеет начальную позицию и целевую позицию. Цель каждого ВС состоит в том, чтобы совершить перелет из начальной позиции в целевую позицию за минимальное время без конфликтов с другими ВС. Конфликт заключается в том, что расстояние между двумя ВС меньше минимального безопасного расстояния (обычно 5 морских миль (м. миль)). В момент времени t положение и курсовой угол ВС записываются как $(x_n(t), y_n(t), \varphi_n(t))$.

Действие для перехода в новое состояние – новое положение, в которое ВС летит из своего текущего положения.

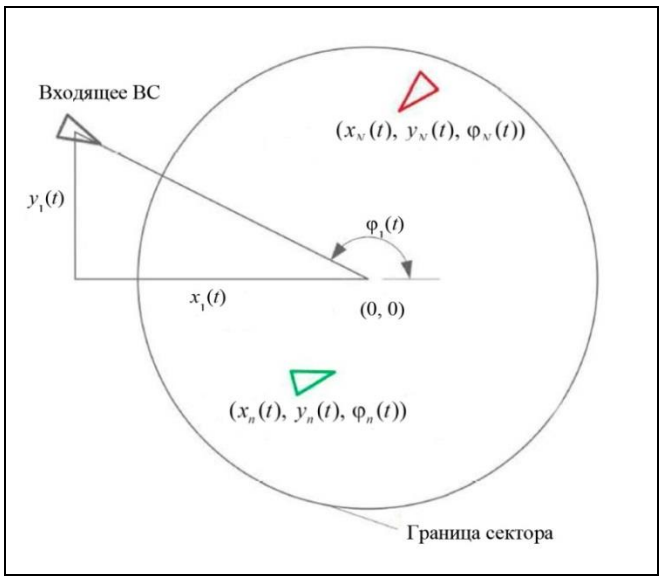

Рис. 1. Сценарий воздушного движения

Тренировочный процесс для одного эпизода показан на рис. 2.

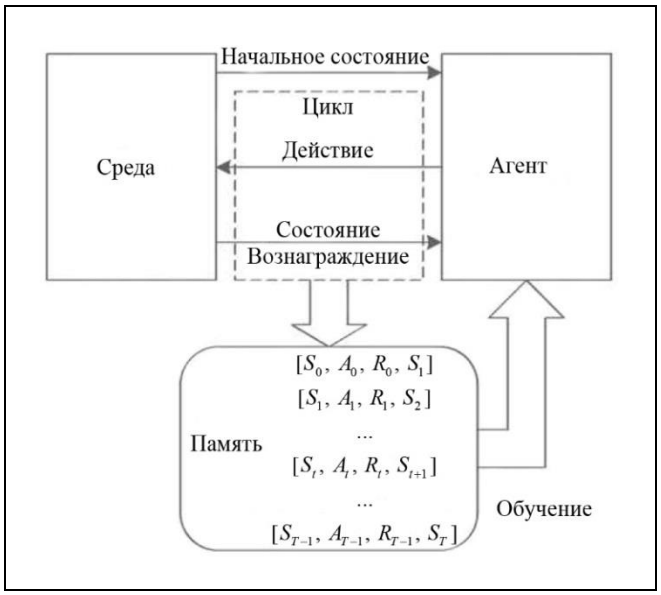

Рис. 2. Схема процесса обучения

В начале каждого эпизода обучения в среде инициализируется состояние S_0. На каждом временном шаге t агент получает состояние S_t и реализует действие A_t. После этого среда переходит в состояние S_{t+1} и возвращает вознаграждение R_t. Этот процесс повторяется до тех пор, пока S_{t+1} не станет конечным состоянием. Наборы значений $[S_t, A_t, R_t, S_{t+1}]$ сохраняются в памяти. Агент выбирает данные из памяти и обучается в соответствии с алгоритмом.

Возможные маневры включают повороты, вертикальную регулировку и изменение скорости. Выходное состояние среды включает в себя положение каждого ВС в секторе. Эксперимент по управлению одним ВС предназначен для проверки того, что окружающая среда может быть использована для обучения агентов. Высота и скорость в этом эксперименте фиксированы.

Набор состояний – это N-мерный вектор, N определяется как $N = N_n \times N_p \times N_d$, где N_n – количество ВС, включая входящий самолет и ВС, находящиеся в секторе; N_p – количество путевых точек ВС, включая все путевые точки от текущего положения до пункта назначения; N_d – размерность местоположения ВС, значение которой равно трем. Размерность высоты фиксирована, а две другие размерности изменяемы. На каждом временном шаге агент воспринимает вектор состояния и после нормализации принимает его за вход нейронных сетей.

Набор действий определяется как

$$A = \left\{\rho, \varphi \,|\, \rho \in [0, L],\ \varphi \in [-\pi, \pi]\right\},$$

где L – радиус сектора; ρ и φ – полярный радиус и угол соответственно. Действие – это позиция, которая описывается двумерной полярной координатой. На каждом временном шаге t агент выбирает действие $A_t \in A$. В зависимости от этого действия входящий самолет перелетает из своего текущего положения в положение A_t.

Целью агента является максимизация долгосрочного вознаграждения и обновление параметров нейронных сетей в соответствии с немедленным вознаграждением. Для создания функции вознаграждения используются четыре правила: отсутствие конфликта между ВС; минимальное время управления; минимальная величина изменения курсового угла; минимальное расстояние полета.

Функция вознаграждения определяется следующим образом:

$$R_t = \begin{cases} -1, & \text{если существует конфликт,} \\ \left|\Delta\varphi_t / \pi\right|, & \text{в противном случае,} \end{cases}$$

где $\Delta\varphi_t \in [-\pi, \pi]$ – изменение курсового угла на шаге t; $\Delta\varphi$ – изменение угла направления на каждом шаге в диапазоне от $-\pi$ до π. Таким образом, $\left|\frac{\Delta\varphi_t}{\pi}\right| \le 1$, что означает, что предотвращение конфликтов имеет наивысший приоритет. Поскольку изменение угла направления приведет к изменению расстояния, расстояние не учитывается в функции вознаграждения.

Действие – это определение полярной координаты, в которой центр сектора задается как полюс, а длина, меньшая радиуса сектора, задается как полярный радиус. Четыре выходных сигнала нейронной сети агента $\mu_\rho, \sigma_\rho, \mu_\varphi, \sigma_\varphi$, представляют среднее значение и стандартное отклонение полярного радиуса и полярного угла. В процессе обучения радиус и угол выбираются в соответствии с нормальным распределением: $\rho \sim N(\mu_\rho, \sigma_\rho)$, $\varphi \sim N(\mu_\varphi, \sigma_\varphi)$ и генерируется двумерное действие. После того, как нейронная сеть агента хорошо обучена, μ_ρ и μ_φ принимаются за ρ и φ.

Обучение нейронных сетей осуществляется следующим образом. Для сети критика определяется параметр δ для оценки выбранного действия:

$$\delta_t = R_t + \gamma \hat{V}(S_{t+1}, w) - \hat{V}(S_t, w),$$

где R_t – немедленное вознаграждение; $\hat{V}(S_t, w)$ – оценочное значение текущего состояния; $\hat{V}(S_{t+1}, w)$ – оценочное значение следующего состояния.

Для обновления параметров w применяется метод наименьших квадратов:

$$w \leftarrow w + \alpha^w \nabla \delta^2.$$

Для сети исполнителя применяется метод градиента политики. Для политики используется уравнение

$$\ln \pi(\rho_t, \varphi_t | S_t, \theta) = \ln \pi(\rho_t | S_t, \theta) + \ln \pi(\varphi_t | S_t, \theta),$$

где $\ln \pi(\rho_t, \varphi_t | S_t, \theta)$ – вероятность выбора ρ и φ в состоянии S_t с параметрами θ; $\ln \pi(\rho_t | S_t, \theta)$ – вероятность выбора ρ в состоянии S_t с параметрами θ; $\ln \pi(\varphi_t | S_t, \theta)$ – вероятность выбора φ в состоянии S_t с параметрами θ. Параметры θ обновляются следующим образом:

$$\theta \leftarrow \theta + \alpha^\theta \delta_t \nabla \ln \pi(\rho_t, \varphi_t | S_t, \theta).$$

Эффективность предложенного подхода продемонстрирована с помощью численного моделирования. Утверждается, что хорошо обученный агент может генерировать решение за время <200 мс, в то время как предыдущие методы требуют десятков или даже сотен секунд для расчета. Кроме того, учитывается радиус поворота ВС, что соответствует реалистичным ситуациям.

2. ГИПЕРЭВРИСТИЧЕСКИЙ ПОДХОД ДЛЯ СНИЖЕНИЯ СЛОЖНОСТИ ВОЗДУШНОГО ДВИЖЕНИЯ [24]

В работе [24] рассматривается задача улучшения структуры воздушного пространства путем минимизации сложности воздушного движения (см. *раздел 2.6*).

Гиперэвристический подход не просто следует определенной метаэвристике, но предполагает гибкую интеграцию и адаптивное управление низкоуровневыми эвристиками. Несколько исследований подтвердили эффективность применения *Q*-обучения в качестве алгоритма обучения для выбора подходящей низкоуровневой эвристики в точке принятия решения. *Q*-обучение происходит путем оценки наилучшей пары «состояние – действие» с использованием памяти на основе *Q*-таблицы. Каждое значение в такой таблице выражает долгосрочную ценность выбора конкретного действия в определенном состоянии.

Традиционная гиперэвристическая структура выбора состоит из двух уровней: первый, который содержит представление проблемы, функцию оценки и набор низкоуровневых эвристик; и второй, который выполняет две отдельные задачи. Первая задача – выбор низкоуровневой эвристики и применение ее к решению. Вторая задача – решение о принятии или отклонении нового решения. Выбор подходящей эвристики и метода принятия решений является нетривиальной задачей при разработке надежной гиперэвристической модели. Гиперэвристика работает без необходимости какой-либо информации о функциональности низкоуровневых эвристик, но обеспечивается полезная обратная связь, позволяющая получить коэффициент использования каждой эвристики и изменение целевой функции. Эта информация имеет решающее значение для процесса обучения.

Как показано на рис. 3, алгоритм использует агент *Q*-обучения для выбора эвристического оператора или низкоуровневой эвристики, которая затем будет применена для генерации решения-кандидата и пересчета его производительности в среде моделирования.

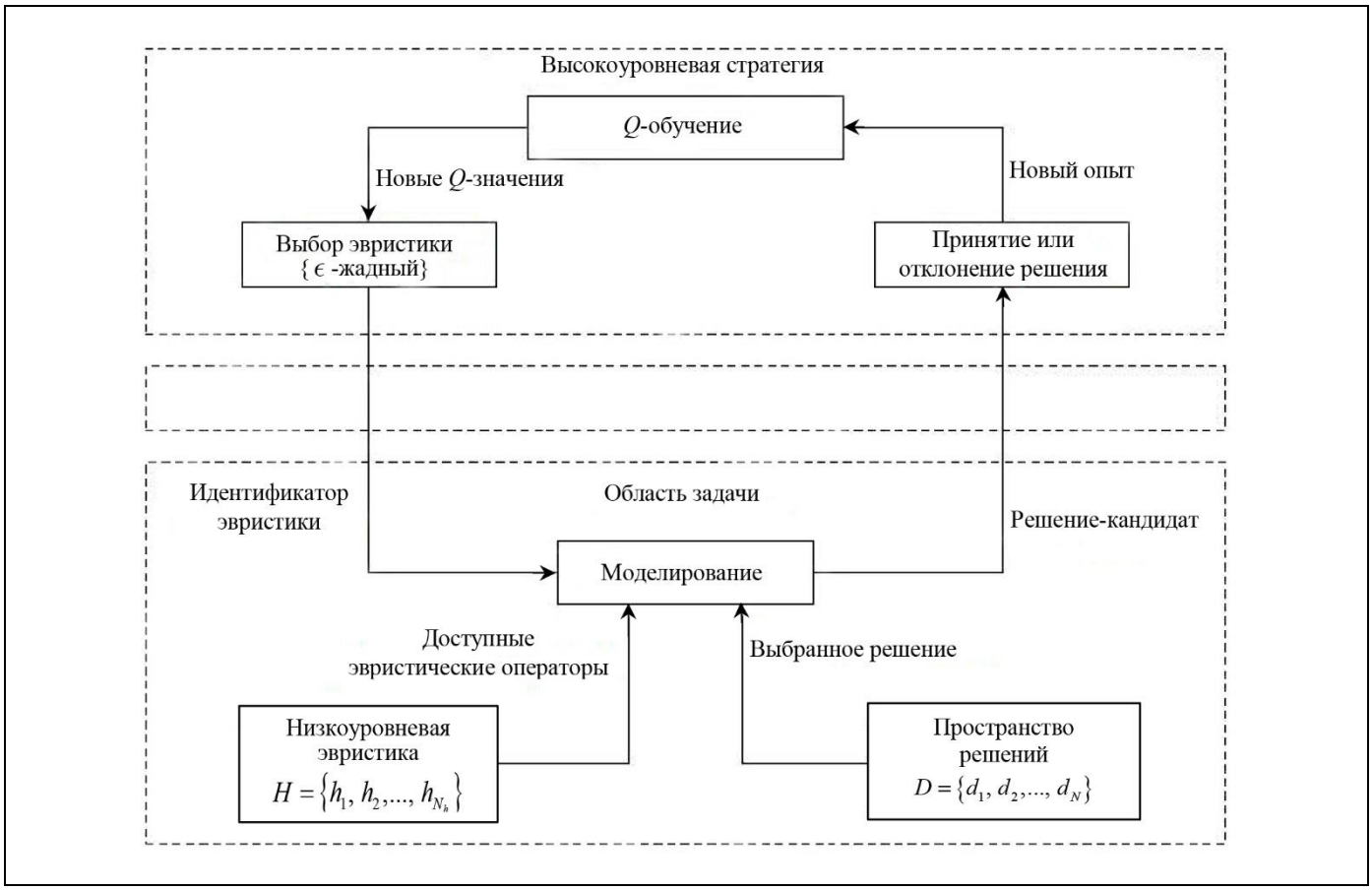

Рис. 3. Структура гиперэвристики, основанной на *Q*-обучении

После этого принимается решение относительно решения-кандидата. Если решение-кандидат может улучшить эффективность, текущее решение будет обновлено. В противном случае решение-кандидат также будет обновлено с некоторой вероятностью, которая по мере обучения будет уменьшаться. Если решение-кандидат принято не будет, то все изменения в пространстве решений, связанные с таким решением-кандидатом, будут отменены при помощи операции возврата. Агент *Q*-обучения определяет вознаграждение, проверяя текущее состояние, выбранный оператор, эволюцию решений и обновляет *Q*-значения в *Q*-таблице.

Предлагаются следующие эвристические операторы (рис. 4).

• h_1 – локальный поиск, который состоит из случайного изменения времени отправления не более чем на 5 мин (рис. 4, *а*);

• h_2 – локальный поиск, который переворачивает текущие путевые точки в горизонтальной плоскости *XY* вдоль траектории (рис. 4, *б*);

• h_3 – локальный поиск, который переворачивает текущие путевые точки в горизонтальной плоскости *XY* перпендикулярно к траектории (рис. 4, *в*);

• h_4 – локальный поиск, который случайным образом выполняет отклонение маршрута путем изменения положения каждой существующей путевой точки (рис. 4, *г*);

• h_5 – оператор, который стремится случайным образом выбрать соседний уровень высоты полета при ограничении максимально допустимых сдвигов уровня полета (рис. 4, *д*).

Для диверсификации во время оптимизации используются следующие три процедуры генерации.

• h_6 – оператор генерации, который случайным образом изменяет время отправления в пределах максимально допустимой смены времени отправления;

• h_7 – оператор генерации, который случайным образом добавляет/удаляет одну или несколько путевых точек в соответствии с ограничениями задачи;

• h_8 – оператор генерации, который стремится случайным образом изменить уровень высоты полета в пределах максимального сдвига уровня высоты полета.

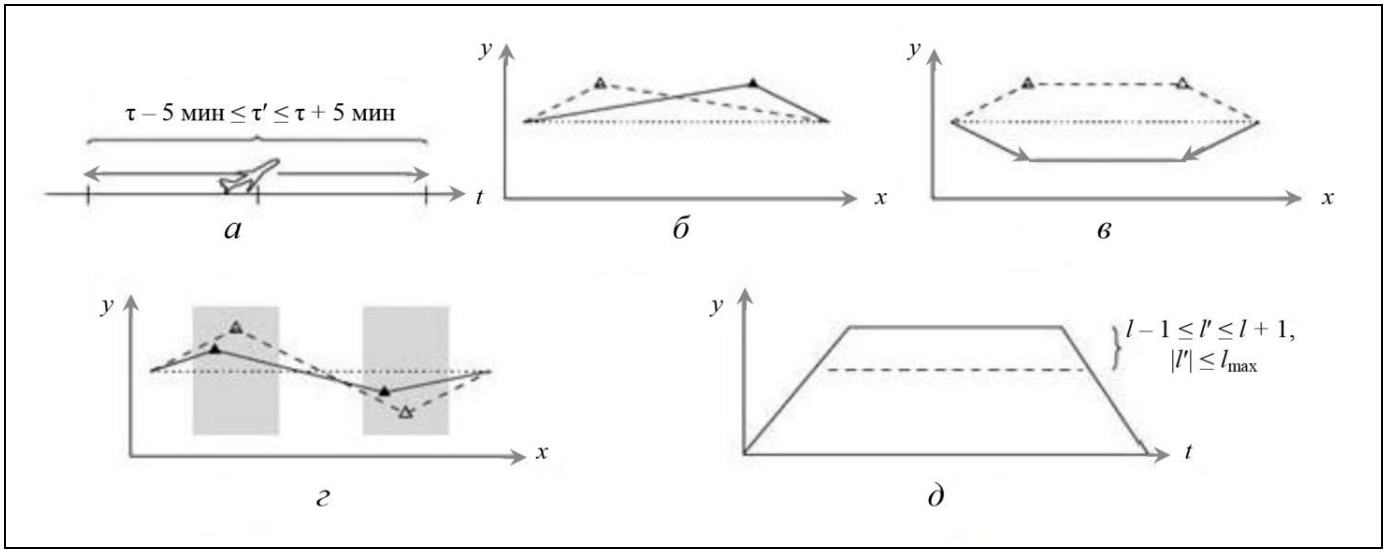

Рис. 4. Представление эвристических операторов: *а* – h_1, *б* – h_2, *в* – h_3, *г* – h_4, *д* – h_5

Q-обучающий агент выполняет ϵ-жадный подход для выбора эвристического оператора на основе *Q*-таблицы. Случайное действие выбирается с вероятностью ϵ, а действие на основе *Q*-таблицы выбирается с вероятностью $1-\epsilon$. Сначала значение ϵ устанавливается на максимальном уровне $\epsilon_{\max}$ (определяется пользователем) и уменьшается в процессе обучения до тех пор, пока не достигает минимального значения $\epsilon_{\min}$ (определяется пользователем).

Цикл обучения начинается с выбора одного из операторов диверсификации h_6, h_7 или h_8. Каждое состояние определяется на основе ранее примененного типа оператора.

Рассматривается семь состояний:

- s_0 – ранее был применен один из операторов диверсификации h_6, h_7 или h_8;
- s_1 – ранее был применен оператор h_1;
- s_2 – ранее был применен оператор h_2;
- s_3 – ранее был применен оператор h_3;
- s_4 – ранее был применен оператор h_4;
- s_5 – ранее был применен оператор h_5;
- s_6 – ранее были применены два последовательных оператора из множества $\{h_1, h_2, h_3, h_4, h_5\}$ без изменения текущего решения.

Q-таблица инициализируется *Q*-значениями согласно табл. 1.

Некоторые *Q*-значения устанавливаются равными 1, чтобы гарантировать, что конкретный эвристический оператор может быть выбран, другие имеют значение 0, чтобы обеспечить переход из одного состояния в другое.

На каждой итерации система сохраняет опыт, содержащий текущие состояния с выбранным оператором *h* и выигрыш *g*, который представляет собой разницу в стоимости между новым решением и предыдущим. В конце цикла агент *Q*-обучения определяет вознаграждения для обновления *Q*-значений в *Q*-таблице.

Q-значения на шаге *t* для найденной пары «состояние – действие» будут обновлены с помощью формулы

$$Q(s_t, h_t) = (1-\alpha)Q(s_t, h_t) + \alpha\left(r_t + \gamma \max_{h\in H} Q(s_{t+1}, h)\right),$$

где $\alpha \in [0, 1]$ обозначает скорость обучения; $\gamma \in [0, 1]$ – коэффициент дисконтирования; r_t – немедленное вознаграждение; $H = \{h_i,\ i = 1,\ldots,8\}$ – множество эвристических операторов.

Принятие перемещения определяет, следует ли принимать или отклонять новое решение на каждом этапе процесса поиска. Итерации продолжаются до тех пор, пока не будет выполнен критерий завершения.

В работе [24] представлена эмпирическая оценка предложенного подхода на основе эксперимента с данными за полный день маршрутного движения во французском воздушном пространстве из 8836 траекторий (рис. 5).

Результат сравнения начальной сложности для первоначального плана траекторий и конечной сложности для рассчитанного при помощи предлагаемого алгоритма окончательного плана траекторий движения в воздушном пространстве Франции для полного дня представлен на рис. 6.

Таблица 1

Инициализация *Q*-таблицы

Состояние	Оператор							
	h_1	h_2	h_3	h_4	h_5	h_6	h_7	h_8
s_0	1	1	1	1	1	–	–	–
s_1	0	1	1	1	1	–	–	–
s_2	1	0	1	1	1	–	–	–
s_3	1	1	0	1	1	–	–	–
s_4	1	1	1	0	1	–	–	–
s_5	1	1	1	1	0	–	–	–
s_6	–	–	–	–	–	1	1	1

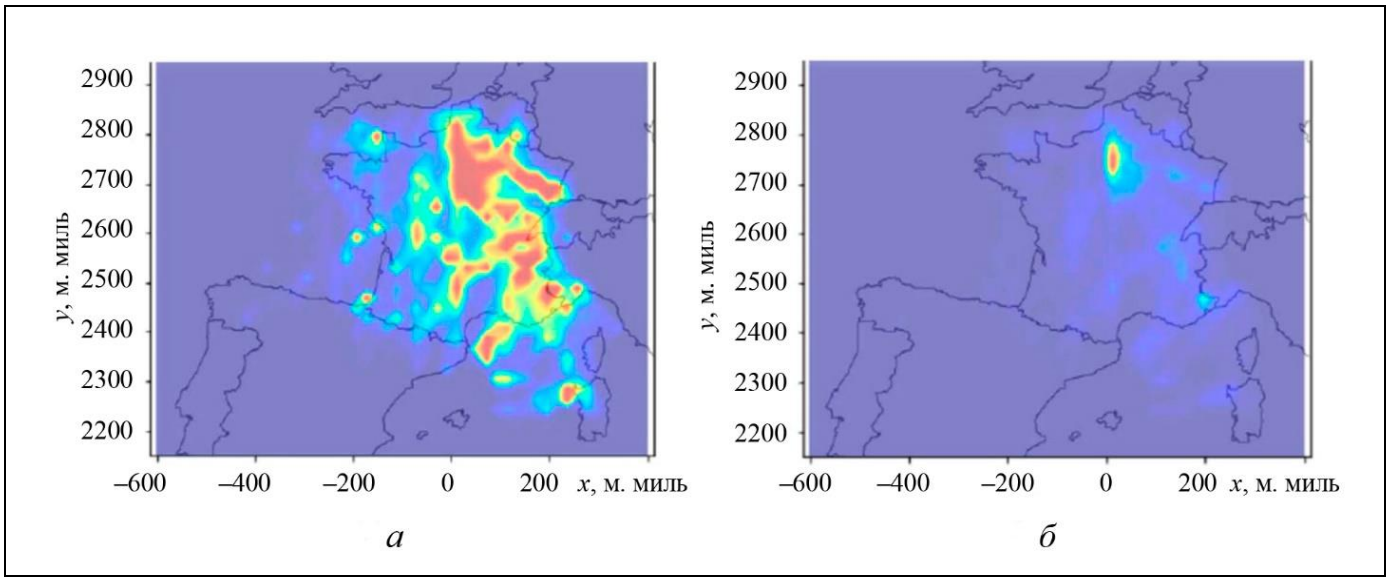

Рис. 5. Карта сложности: *а* – для начальных траекторий и *б* – для конечных траекторий полного дня движения во французском воздушном пространстве

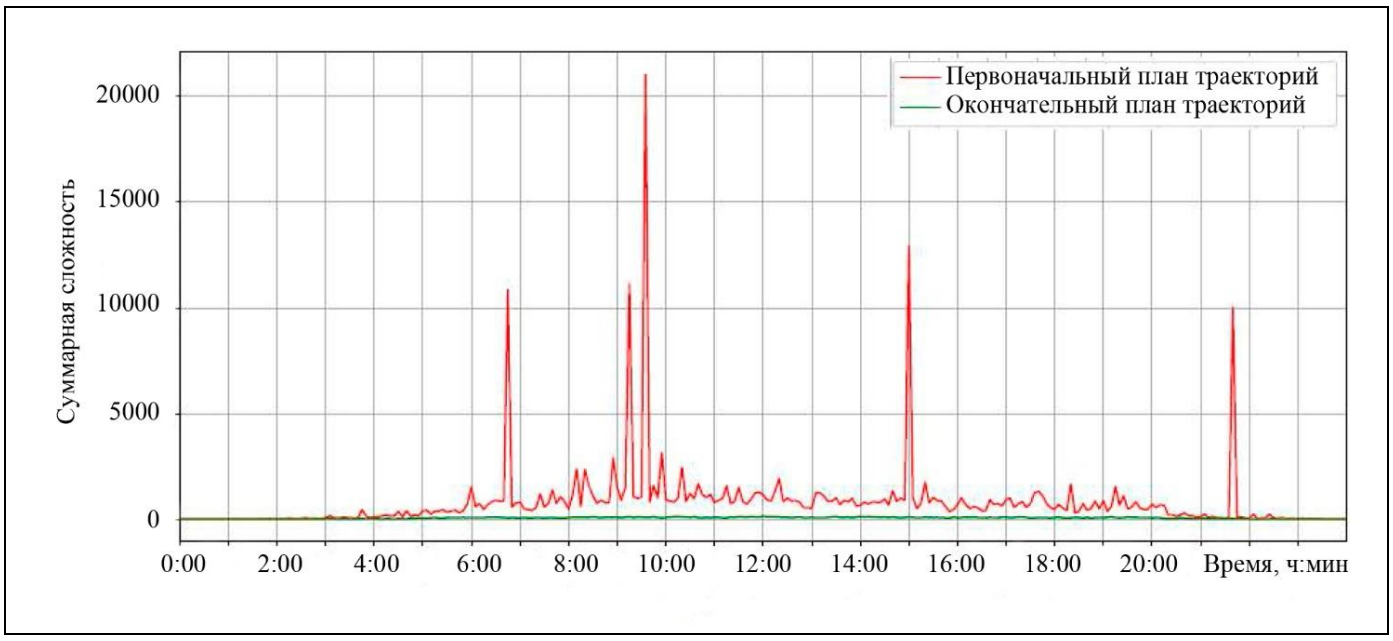

Рис. 6. Сравнение начальной сложности и конечной сложности для полного дня движения в воздушном пространстве Франции

3. ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ ИСПОЛЬЗОВАНИЯ ВЗЛЕТНО-ПОСАДОЧНЫХ ПОЛОС С ПРИМЕНЕНИЕМ ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ

Взлетно-посадочные полосы аэропортов являются одним из основных узких мест в воздушном движении и одним из ключевых факторов, определяющих пропускную способность аэропорта. Строительство новой взлетно-посадочной полосы не всегда возможно. Одним из подходов к решению проблемы пропускной способности является модернизация структуры воздушного пространства и инфраструктуры аэродромов. В работе [25] рассматривается реализация такого подхода при помощи математического моделирования. В настоящем обзоре рассматривается другой подход, заключающийся в оптимизации использования инфраструктуры с помощью улучшенного планирования посадок ВС.

Решение задачи оптимизации посадок состоит из трех шагов: сначала создается первоначальное расписание в порядке живой очереди. Затем оно модифицируется во время так называемой фазы захода на посадку и, наконец, замораживается, когда самолет достигает конечной фазы захода на посадку. Первоначальное расписание включает ВС, находящиеся в зоне действия радара посадочного устройства аэропорта, что соответствует временно́му горизонту примерно 40 мин до посадки. Процесс обновления выполняется каждый раз, когда новый самолет входит в зону действия радара, чтобы улучшить расписание посадки [26].

Наиболее распространенные требования включают в себя безопасное разделение между последовательными ВС, разрешенные временны́е интервалы, определяемые самым ранним и самым поздним временем полета на основе расхода топлива, и ограничения приоритета. Различные целевые функции предназначены для увеличения пропускной способности взлетно-посадочной полосы, соблюдения расписания, минимизации расхода топлива и т. д.

Известно, что задача оптимизации очереди и времен посадок является *NP*-трудной [27]. Следовательно, время решения точными методами быстро увеличивается с ростом количества ВС. С момента публикации первого подхода к решению [28] в 1976 г. до настоящего времени в литературе появилось несколько моделей и подходов к решению, в том числе генетические и эвристические алгоритмы решения задачи, которые позволяют за приемлемое время получить, возможно, не оптимальное, но достаточно эффективное решение [29, 30]. Обзор [31] посвящен некоторым точным подходам к решению задачи (в основном смешанному целочисленному программированию), а в обзоре [32] рассматриваются методы приближенного решения задачи, в основном генетические и меметические алгоритмы. В последнее время появился новый многообещающий подход к решению задачи, основанный на обучении с подкреплением.

В работе [33] рассматривается проблема планирования взлетов ВС на одной взлетно-посадочной полосе с целью соблюдения установленных временных интервалов взлета. Проблема моделируется как марковский процесс принятия решений и решается с применением алгоритма *Q*-обучения [34]. Агенты соответствуют ВС, состояния соответствуют положению ВС на земле в зависимости от его фазы (стоянка, руление, взлет). Действие заключается в задержке ВС, а вознаграждение определяется таким образом, чтобы свести к минимуму задержку во время руления и соблюдать установленные для ВС временные интервалы. Тестирование проводилось на реальных данных из международного аэропорта имени Джона Ф. Кеннеди, которые включали вылеты 698 рейсов, что соответствует двум дням работы в этом аэропорту. На основе этих данных сгенерированы 42 сценария обучения. Результаты показывают, что алгоритм имеет производительность, аналогичную производительности авиадиспетчеров или превышающую ее.

В работе [35] предлагается структура для моделирования проблемы упорядочивания и разделения ВС, чтобы модель соответствовала среде NASA sector-33 [36], которая представляет собой приложение для управления воздушным движением, содержащее 35 примеров задач с участием до пяти самолетов. Они включают в себя обеспечение контроля скорости и маршрута для ВС.

Предлагаемая модель состоит из агентов, состояний, действий и вознаграждений. Эта модель включает в себя два типа агентов: родительские агенты и дочерние агенты. Состояние родительского агента содержит снимок игрового экрана. Состояние дочернего агента содержит информацию о цели фиксации измерения, скорости и ускорении ВС, идентификаторе маршрута в дополнение к информации об N ближайших агентах, чтобы разрешить связь между агентами. Действия для

родительского/дочернего агента заключаются в изменении или поддержании маршрута/скорости ВС. Вознаграждение разработано таким образом, чтобы наказывать конфликтующих агентов (разделенных менее чем на 3 м. мили).

Затем для решения модели применяется иерархический алгоритм глубокого обучения с подкреплением. Этот алгоритм сочетает в себе алгоритм Q-обучения [18] и нейронные сети [5]. Он называется иерархическим, потому что действия выполняются на двух уровнях: на родительском уровне выбирается маршрут, затем на дочернем уровне выбирается скорость для ВС. Тесты, выполненные в вышеуказанной среде NASA (с участием от двух до пяти ВС) показывают жизнеспособность предлагаемого подхода.

3.1. Математическая модель

Более подробно остановимся на подходе, рассматриваемом в работе [37]. Предложен распределенный алгоритм, основанный на Q-обучении, параметры которого оптимально настроены при помощи генетического алгоритма. Алгоритм реализован с помощью механизма скользящего временно́го окна.

Рассматривается граф $G(N, L)$, где N – множество вершин, а L – множество ребер. Среди множества вершин можно выделить два подмножества: $N_e \subset N$ – подмножество точек входа в TMA (Terminal Maneuvering Areas); $N_r \subset N$ – подмножество взлетно-посадочных полос. Конечные звенья, соединяющие взлетно-посадочные полосы, также сгруппированы в подмножество звеньев: $L_r \subset L$.

На рис. 7 рассматривается сеть в парижском аэропорту имени Шарля де Голля (CDG). Воздушное судно входит в точки LORNY или OKIPA, используются две взлетно-посадочные полосы 27R и 26L, точки слияния IF_27R и IF_26L. Используется двухточечная система слияния (IF_27R – RWY_27R и IF_26L – RWY_26L соответственно).

Для каждого ВС процедура выполняется с постоянной скоростью. Эта скорость является посадочной скоростью, зависящей от категории вихревого следа.

Структура PMS (Point Merge System) представлена на рис. 8. Для каждого ВС длина дуги PMS будет рассматриваться в качестве переменной решения для алгоритма.

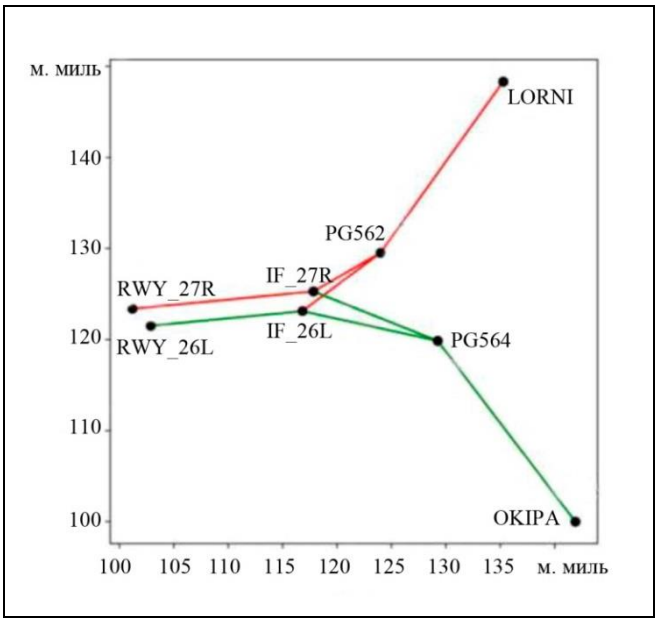

Рис. 7. Рассматриваемая сеть в аэропорту имени Шарля де Голля

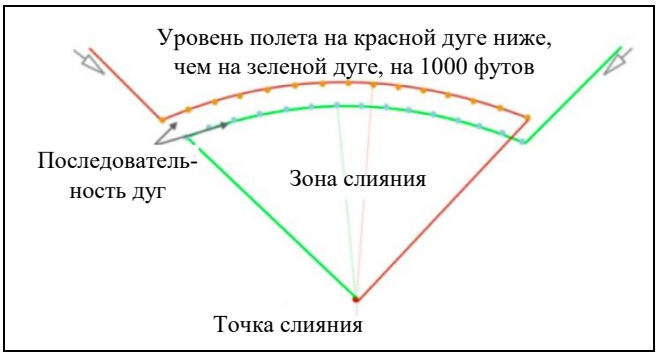

Рис. 8. Топология точек слияния: для каждого самолета длина дуги последовательности полетов является переменной решения

Рейс f имеет следующие характеристики:

- $V_{0,f}$ – начальная истинная скорость ВС;
- $t_{0,f}^{\text{TMA}}$ – начальное время входа в TMA;
- $r_{0,f} \in N_r$ – взлетно-посадочная полоса, на которую планируется посадка ВС;
- t_f^{RTA} – время, в которое ВС должно приземлиться (англ. *Required Time of Arrival*, RTA);
- C_f – категория вихревого следа.

Для каждого рейса f рассматриваются следующие переменные решения:

- V_f – скорость ВС;
- t_f^{TMA} – время входа в TMA;
- r_f – взлетно-посадочная полоса, назначенная для посадки;
- l_f^{MP} – длина дуги точки слияния.

Скорость ВС должна оставаться в заданном диапазоне начальной скорости:

$$V_f \in V_{0,f} + p\Delta V,$$

где p – количество приращений, а ΔV – приращение скорости:

$$p \in Z,\ p\Delta V \in \left[\Delta V^{\min}, \Delta V^{\max}\right].$$

Здесь $\Delta V^{\max}$ – максимальное увеличение скорости от $V_{0,f}$ которое может быть присвоено ВС; $\Delta V^{\min}$ – минимальное снижение скорости ВС от $V_{0,f}$, которое зависит от категории вихревого следа.

Решение о времени входа соответствует задержке, которая возможна в воздушном пространстве на маршруте до того, как ВС войдет в TMA. В этом воздушном пространстве ВС может замедляться или ускоряться в заданном диапазоне. В результате, время входа ВС в TMA также может изменяться в заданном диапазоне:

$$t_f^{\text{TMA}} \in t_{0,f}^{\text{TMA}} + p\Delta T,$$

где p – количество приращений, а ΔT – приращение по времени:

$$p \in Z,\ p\Delta T \in \left[\Delta T^{\min}, \Delta T^{\max}\right].$$

Здесь $\Delta T^{\max}$, $\Delta T^{\min}$ – максимальное и минимальное приращения времени от $t_{0,f}^{\text{TMA}}$, которые могут быть заданы ВС.

Для поддержания сбалансированного потока между взлетно-посадочными полосами иногда может быть более целесообразным изменить посадочную полосу ВС $r_f \in N_r$.

Поскольку сеть содержит точки слияния, одна из переменных принятия решения, l_f^{MP}, является длиной дуги, по которой ВС будет лететь в одну из точек слияния.

$$l_f^{\text{MP}} \in p\Delta L,$$

где p – число приращений, а ΔL – приращение длины:

$$p \in N,\ p\Delta L \le L_{MP}^{\max},$$

и $L_{\text{MP}}^{\max}$ – максимальная длина дуги до точки слияния.

3.2. Описание алгоритма глубокого обучения [37]

В настоящем разделе описывается алгоритм глубокого обучения на основе модели, представленной в разделе 3.1, предназначенный для разрешения потенциальных конфликтов между ВС при сильной загрузке в зоне аэропорта за приемлемое время.

Каждый полет представляет собой марковский процесс принятия решений $MDP\{S, A, P_a, R_a\}$. Все переменные решения представляют пространство состояний S. Это означает, что для каждого ВС состояние определяется {скоростью, временем входа в TMA, длиной дуги PMS, назначением взлетно-посадочной полосы}. В каждом состоянии рассматриваются следующие действия: A = {увеличение/уменьшение скорости, увеличение/уменьшение времени входа в TMA, увеличение/уменьшение длины дуги PMS, изменение посадочной полосы, никаких действий}. Для состояний, которые не являются прямыми соседями для текущего состояния, значение функции перехода равно нулю. Для соседних состояний функция перехода является равносбалансированной:

$$P_a(s, s') = \begin{cases} 0, \text{ если } s \text{ и } s' - \text{не соседние}, \\ \dfrac{1}{Card(A)}, \text{ в противном случае}, \end{cases}$$

где $Card(A)$ – количество элементов в A, в данном случае восемь.

Q-обучение – это алгоритм обучения с подкреплением без модели. Это означает, что алгоритму не нужна модель среды, он только взаимодействует со средой, не зная ее. Каждое ВС рассматривается как агент, что делает алгоритм мультиагентным.

Q-обучение используется для изучения оптимальной политики марковского процесса принятия решений. Это делается путем вычисления Q-функции для каждого ВС, представляющей ожидаемое вознаграждение, которое может получить агент, если он предпримет данное действие в данном состоянии. Используемое Q-обучение является распределенным, что означает, что вознаграждение каждого агента обрабатывается индивидуально на каждой итерации.

Для каждого агента ожидаемое вознаграждение $Q(s, a)$ в заданном состоянии s за заданное действие a обновляется следующим образом:

$$Q(s, a) = Q(s, a) + \alpha\left(R + \gamma \max_{a'} Q(s', a') - Q(s, a)\right),$$

где s' – новое состояние, когда действие a выполняется в состоянии s; R – вознаграждение, которое агент получит, совершив действие a в состоянии s; α – скорость обучения; γ – коэффициент дисконтирования.

Ожидаемое вознаграждение $Q(s, a)$ в данном состоянии s за данное действие a обновляется на каждой итерации с учетом оценки из оптимального будущего значения $\max\limits_{a'} Q(s', a')$. Это делается независимо от проводимой политики. Это одноступенчатый алгоритм, поскольку оценка выполняется только путем просмотра на одну итерацию вперед.

Для состояния $s \in S$, действия $a \in A$ и параметра T, называемого температурой, вероятность $\pi(s, a)$ выбрать действие a в состоянии s определяется как

$$\pi(s, a) = \frac{e^{Q(s,a)/T}}{\sum\limits_{a' \in A} e^{Q(s,a')/T}}.$$

Температура при итерации k задается при помощи параметра β следующим образом: $T_k = T_0 \beta^k$, где T_0 – начальная температура. Эта температура устанавливает компромисс между исследованием и использованием: относительно высокая температура будет способствовать формированию таблицы Q, низкая температура – использованию таблицы Q.

В этом распределенном Q-обучении каждый самолет рассматривается как обучающийся агент и, следовательно, имеет Q-таблицу. Все Q-таблицы инициализируются при значении Q_0, выбранном относительно низким для принудительного изучения состояния. Это делается специально, поскольку вознаграждение и Q-таблица самолета зависят от агентов, близких к нему, и они могут конфликтовать. Каждый агент рассматривается как независимый ученик и при выборе своего действия не учитывает выбранное другими агентами действие, а только их фактические состояния. Следовательно, в промежутке между двумя решениями агента его окружение может быть изменено. Агент может выбрать действие «ничего не делать» и тогда его состояние не изменится.

Для каждого ВС вычисляется функция вознаграждения, которая будет использоваться алгоритмом обучения с подкреплением. Вознаграждение, выдаваемое за каждое состояние и действие, зависит от состояния других ВС и рассчитывается как взвешенная сумма вознаграждений, описанных далее.

Все вознаграждения являются отрицательными (штрафами):

$$R = \omega_{\text{RTA}} \left(R_{\text{RTA}} + 5R_{runway} \right) + \omega_{conflict} (\sum R_{link} + \sum R_{node}).$$

Если самолет f не приземляется на $r_{0,f}$, его предпочтительной взлетно-посадочной полосе, добавленная награда в пять раз превышает значение R_{runway}, взвешенное на ω_{RTA}; ω_{RTA}, $\omega_{conflict}$ – параметры алгоритма.

Далее описываются различные части функции вознаграждения.

• Требуемое время прибытия. У всех авиакомпаний есть расписание для каждого ВС, и ВС, прилетевшее вовремя, должно получать более высокое вознаграждение. К вознаграждению каждого ВС добавляется значение в соответствии с реальным временем прибытия $t_{arrival}$:

$$R_{\text{RTA}} = -\left| t_f^{\text{RTA}} - t_{arrival} \right|.$$

• Номер взлетно-посадочной полосы

$$R_{runway} = \begin{cases} 0, & \text{приземлился на } r_{0,f}, \\ -1, & \text{в противном случае.} \end{cases}$$

• Конфликты. Модель рассматривает два вида конфликтов: конфликт на ребре, когда два ВС не соблюдают разделение категорий вихревого следа, и конфликт на вершине, когда ВС не соблюдают горизонтальное разделение в точках слияния [38]. Во избежание конфликта для каждого ребра на входе и выходе минимальное расстояние между двумя ВС f и g должно соответствовать табл. 2.

Предполагая, что $s_{f,g}$ – минимальный интервал, а $d_{f,g}$ – фактическое расстояние между ведущим самолетом f и замыкающим g (рис. 9), критичность потенциального конфликта, C_{link}, пропорциональна расстоянию между ВС. Также вычисляются обгоны; если это происходит, то $d_{f,g} < 0$ и критичность конфликта устанавливается равной –1:

$$C_{link} = \begin{cases} -1, \text{ если } d_{f,g} < 0, \\ -\dfrac{\left| s_{f,g} - d_{f,g} \right|}{s_{f,g}}, \text{ если } d_{f,g} < s_{f,g}, \\ 0, \text{ в противном случае.} \end{cases}$$

Таблица 2

Минимальное расстояние между ВС, м. миль

Категория ВС		Ведущее ВС f		
		Тяжелое	Среднее	Легкое
Ведомое ВС g	Тяжелое	4	3	3
	Среднее	5	3	3
	Легкое	6	5	3

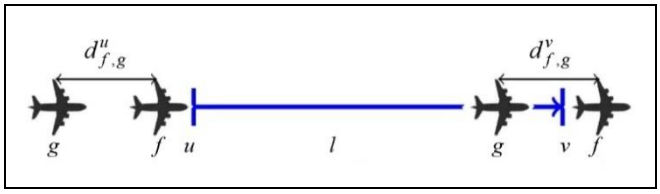

Рис. 9. Обнаружение конфликта между ВС

C_{link} – это кусочно-линейная и непрерывная функция, которая необходима для алгоритма обучения, чтобы знать, усугубляется конфликт или нет. Поскольку C_{link} может быть близок к нулю, алгоритм обучения может улучшить R_{RTA} вместо разрешения конфликта. Чтобы расставить приоритеты в решении конфликта, значение функции вознаграждения искусственно устанавливается в диапазоне от –0,3 до –1 по формуле

$$R_{link} = -0{,}3 + \left(C_{link} \cdot (1 - 0{,}3)\right).$$

Если нет конфликта на ребрах между двумя ВС f и g, могут быть конфликты на вершинах. В ТМА каждое ВС должно быть отделено от других на 3 м. мили, чтобы соблюдать дистанцию разделения. Но, как показано в работе [38], во многих аэропортах, благодаря геометрии сети, зона обнаружения может быть уменьшена до круга радиусом 2,2 м. мили. Критичность конфликта на вершине определяется формулой

$$C_{node} = \begin{cases} -\dfrac{2{,}2 - d_{f,g}}{2{,}2}, \text{ если } d_{f,g} < 2{,}2, \\ 0, \text{ в противном случае.} \end{cases}$$

Значение функции вознаграждения, используемой для конфликта на вершине, искусственно устанавливается между –0,3 до –1 по формуле

$$R_{node} = -0{,}3 + \left(C_{node} \cdot (1 - 0{,}3)\right).$$

В этой задаче, если ВС входит в ТМА за несколько часов до другого, их решения можно рассматривать как независимые. Поэтому динамическая задача оптимизации посадок ВС решается на основе скользящего временного окна.

В процессе оптимизации ВС внутри скользящего временного окна подразделяются на четыре группы:

- ВС, которые уже приземлились;
- ВС, по которым решение о посадке уже принято;
- активные ВС, по которым принимается решение;
- ВС, по которым решение будет приниматься позднее.

На каждой итерации скользящего окна алгоритм оптимизации выполняется на активных рейсах.

Запуск алгоритма на каждом активном рейсе в скользящем окне недостаточно эффективен, некоторые из активных рейсов могут иметь хорошее вознаграждение, а другие ВС могут иметь конфликты. Чтобы ускорить процесс оптимизации, решения для ВС с худшим вознаграждением изменяются с более высоким приоритетом. Эти ВС указываются как критически важные. Они определяются с использованием порогового значения, которое составляет более 70 % от наихудшего вознаграждения. Поскольку ВС обучаются и среднее вознаграждение снижается, то все больше ВС становятся критически важными.

Алгоритм был успешно опробован на данных парижского аэропорта имени Шарля де Голля с искусственно увеличенным до 687 общим количеством посадок ВС. Бесконфликтное решение для полного дня трафика было рассчитано менее чем за 30 с, что приемлемо для планирования в режиме реального времени.

ЗАКЛЮЧЕНИЕ

В течение нескольких десятилетий проводились обширные исследования по проблеме автоматизации поддержки принятия решений в системах управления воздушным движением. Математические модели, разработанные в связи с этой проблемой, основаны либо на минимизации числа потенциальных конфликтов между четырехмерными траекториями ВС, либо на перераспределении потоков ВС с целью уменьшения перегрузки воз-

душного пространства. Для уменьшения числа потенциальных конфликтов между ВС в основном используется одно или несколько из следующих действий: задержки вылета рейсов, регулирование скорости в воздухе, изменение траекторий полета, изменение уровня высоты полета.

Было показано, что задача минимизации числа потенциальных конфликтов между ВС является *NP*-трудной. Это привело к появлению различных метаэвристических алгоритмов для ее решения. Для стратегического планирования потока ВС с учетом неопределенности положения ВС был разработан гибридный метаэвристический подход на основе алгоритма имитации отжига, улучшенный методами локального поиска.

Сложность и масштабность задачи минимизации числа потенциальных конфликтов в воздушном пространстве требует поиска новых подходов к ее решению. В последние годы появились работы, в которых применяются методы глубокого обучения с подкреплением для решения задач, связанных с повышением эффективности и безопасности воздушного движения. Эффективность предлагаемых подходов исследовалась при помощи вычислительных экспериментов, которые показали обнадеживающие результаты. Для оценки перспективности использования предлагаемых подходов в реальной обстановке требуются дальнейшие обширные исследования.

ЛИТЕРАТУРА

1. *Кулида Е.Л., Лебедев В.Г.* Методы решения задач планирования и регулирования потоков воздушного движения. Ч. 1. Стратегическое планирование четырехмерных траекторий // Проблемы управления. – 2023. – № 1. – С. 3–14. [*Kulida, E.L. and Lebedev, V.G.* Methods for Solving Some Problems of Air Traffic Planning and Regulation. Part I: Strategic Planning of 4D Trajectories // Control Sciences. – 2023. – No. 1. – P. 2–11.]
2. *Degas, A., Islam, M.R., Hurter, C.,* et al. A Survey on Artificial Intelligence (AI) and eXplainable AI in Air Traffic Management: Current Trends and Development with Future Research Trajectory // Applied Sciences. – 2022. – Vol. 12, no. 3. – Art. no. 1295. – DOI: 10.3390/app12031295.
3. *Wang, Z., Pan, W., Li, H.,* et al. Review of Deep Reinforcement Learning Approaches for Conflict Resolution in Air Traffic Control // Aerospace. – 2022. – Vol. 9, no. 6. – Art. no. 294. – DOI: 10.3390/aerospace9060294.
4. *Brittain, M., Wei, P.* Autonomous Aircraft Sequencing and Separation with Hierarchical Deep Reinforcement Learning // Proceedings of the 8th International Conference on Research in Air Transportation. – Barcelona, Spain, 2018. – URL: https://www.reseachgate.net/publication/327287314.
5. *Pham, D.T., Tran, N.P., Alam, S.,* et al. A Machine Learning Approach for Conflict Resolution in Dense Traffic Scenarios with Uncertainties // ATM 2019, 13th USA/Europe Air Traffic Management Research and Development Seminar. – Vienne, 2019.
6. *Pham, D.T., Tran, N.P., Alam, S.,* et. al. Deep Reinforcement Learning based Path Stretch Vector Resolution in Dense Trac with Uncertainties // Transportation Research. Part C. Emerging Technologies. – 2021. – Vol. 135. – Art. no. 103463. – DOI: 10.1016/j.trc.2021.103463.
7. *Tran, P.N., Pham, D.T., Goh, S.K.,* et al. An Interactive Conflict Solver for Learning Air Traffic Conflict Resolutions // Journal of Aerospace Information Systems. – 2020. – Vol. 17, no. 6. – P. 271–277.
8. *Ribeiro, M., Ellerbroek, J., Hoekstra J.* Improvement of Conflict Detection and Resolution at High Densities through Reinforcement Learning // Proceedings of the International Conference on Research in Air Transportation. – Tampa, USA, 2020.
9. *Brittain, M., Wei, P.* Autonomous Separation Assurance in an High-Density en Route Sector: A Deep Multi-Agent Reinforcement Learning Approach // IEEE Intelligent Transportation Systems Conference (ITSC). – Aukland, New Zealand, 2019. – P. 3256–3262.
10. *Brittain, M., Yang, X., Wei, P.* A Deep Multi-Agent Reinforcement Learning Approach to Autonomous Separation Assurance // Arxiv:2003.08353v2. – 2020. – DOI: https://doi.org/10.48550/arXiv.2003.08353.
11. *Brittain, M., Wei, P.* One to Any: Distributed Conflict Resolution with Deep Multi-Agent Reinforcement Learning and Long Short-Term Memory // AIAA Scitech 2021 Forum. – Nashville, 2021. – P. 1952.
12. *Zhao, P., Liu, Y.* Physics Informed Deep Reinforcement Learning for Aircraft Conflict Resolution // IEEE Transactions on Intelligent Transportation Systems. – 2021. –Vol. 23, iss. 7. – P. 8288–8301. – DOI: 10.1109/TITS.2021.3077572.
13. *Mollinga, J.; Hoof, H.* An Autonomous Free Airspace En-route Controller Using Deep Reinforcement Learning Techniques // Arxiv:2007.01599. – 2020. – DOI: https://doi.org/10.48550/arXiv.2007.01599.
14. *Khan, N.A., Brohi, S.N., Jhanjhi, N.* UAV's Applications, Architecture, Security Issues and Attack Scenarios: A Survey // Intelligent Computing and Innovation on Data Science. – 2020. – Vol. 183. – P. 753–760. – DOI: 10.1007/978-981-15-3284-9_86.
15. *Szegedy, C., Zaremba, W., Sutskever, I.,* et al. Intriguing Properties of Neural Networks // Arxiv:1312.6199v3. – 2013. – DOI: https://doi.org/10.48550/arXiv.1312.6199.
16. *Athalye, A., Engstrom, L., Ilyas, A., Kwok, K.* Synthesizing Robust Adversarial Examples // International Conference on Machine Learning. – Stockholm, 2018. – P. 284–293.
17. *Wang, L., Yang, H., Lin, Y.,* et al. Explainable and Safe Reinforcement Learning for Autonomous Air Mobility // arXiv:2211.13474v1. – 2022. – DOI: https://doi.org/10.48550/arXiv.2211.13474.
18. *Messaoud, M.* A Thorough Review of Aircraft Landing Operation from Practical and Theoretical Standpoints at an Airport Which May Include a Single or Multiple Runways // Applied Soft Computing. – 2020. – Vol. 98, no. 12. – Art. no. 106853. – DOI: 10.1016/j.asoc.2020.106853.
19. *Sutton, R.S., Barto, A.G.* Reinforcement Learning: An Introduction. – London, UK: MIT Press, 2017.
20. *Degris, T., Pilarski, P.M., Sutton, R.S.* Model-Free Reinforcement Learning with Continuous Action in Practice // American Control Conf., Fairmont Queen. – Elizabeth, Montréal, Canada, 2012. – P. 2177–2182.

21. *LeCun, Y., Bengio, Y., Hinton, G.* Deep Learning // Nature. – 2015. – Vol. 521. – P. 436–444.
22. *Sutton, R.S., McAllester, D.A., Singh, S.P.*, et al. Policy Gradient Methods for Reinforcement Learning with Function Approximation // Advances in Neural Information Processing Systems. – USA, Denver, MIT Press, 1999. – P. 1057–1063.
23. *Wang, Z., Li, H., Wang, J., Shen, F.* Deep Reinforcement Learning Based Conflict Detection and Resolution in Air Traffic Control // IET Intelligent Transport System. – 2019. – Vol. 13. – P. 1041–1047.
24. *Juntama, P., Delahaye, D., Chaimatanan, S., Alam, S.* Hyperheuristic Approach Based on Reinforcement Learning for Air Traffic Complexity Mitigation // Journal of Aerospace Information Systems. – 2022. – Vol. 19, no. 9. – DOI: 10.2514/1.i011048.
25. *Вишнякова Л.В., Попов А.С.* Выбор структуры воздушного пространства и инфраструктуры аэродромов при их модернизации методами математического моделирования // Известия РАН. Теория и системы управления. – 2021. – № 6. – С. 66 – 105. [*Vishnyakova, L.V., Popov, A.S.* Vybor struktury vozdushnogo prostranstva i infrastruktury aerodromov pri ih moder-nizacii metodami matematicheskogo modelirovaniya // Izve-stiya RAN. Teoriya i sistemy upravleniya. – 2021. – No. 6. – P. 66 – 105. (In Russian)]
26. *Bennell, J.A. Mesgarpour, M. Potts, C.N.* Airport Runway Scheduling // Semantic Scholar. – 2011. – Vol. 4OR. – P. 115–138. – DOI: 10.1007/s10288-011-0172-x .
27. *Prakash, R. Piplani, R. Desai, J.* An Optimal Data-Splitting Algorithm for Aircraft Scheduling on a Single Runway to Maximize Throughput // Transportation Research, Part C: Emerging Technologies. – 2018. – Vol. 95. – P. 570– 581.
28. *Dear, R.G.* The Dynamic Scheduling of Aircraft in the Near Terminal Area // Technical Report, R76-9, Flight Transportation Laboratory. – MIT, Cambridge, MA, USA, 1976.
29. *Кулида Е.Л.* Генетический алгоритм решения задачи оптимизации последовательности и времен посадок воздушных судов // Автоматика и телемеханика. – 2022. – № 3. – С. 156–168. [*Kulida, E.L.* Genetic Algorithm for Solving the Problem of Optimizing Aircraft Landing Sequence and Times // Automation and Remote Control. – 2022. – Vol. 83, no. 3. – P. 426–436.]
30. *Кулида Е.Л., Лебедев В.Г., Егоров Н.А.* Сравнение двух алгоритмов решения задачи оптимизации последовательности и времен посадок воздушных судов // В сборнике: Управление развитием крупномасштабных систем (MLSD'2021). Труды Четырнадцатой международной конференции. – Москва. – 2021. – С. 1438–1444. [*Kulida E., Egorov N., Lebedev V.* Comparison of Two Algorithms for Solving the Problem Aircraft Arrival Sequencing and Scheduling / Proceedings of the 14th International Conference «Management of Large-Scale System Development» (MLSD). – IEEE, 2021. – URL: https://ieeexplore.ieee.org/document/9600243.]
31. *Вересников Г.С., Кулида Е.Л., Егоров Н.А., Лебедев В.Г.* Методы построения оптимальных очередей воздушных судов на посадку. Ч. 1. Методы точного решения // Проблемы управления. – 2018. – № 4. – С. 2–14. [*Veresnikov, G.S., Egorov, N.A. Kulida, E.L., Lebedev, V.G.* Methods for Solving of the Aircraft Landing Problem. I. Exact Solution Methods. // Automation and Remote Control. – 2019. – Vol. 80. – P. 1317–1334.]
32. *Вересников Г.С., Егоров Н.А., Кулида Е.Л., Лебедев В.Г.* Методы построения оптимальных очередей воздушных судов на посадку [1]. Ч. 2. Методы приближенного решения // Проблемы управления. – 2018. – № 5. – С. 2–13. [*Veresnikov, G.S., Egorov, N.A. Kulida, E.L., Lebedev, V.G.* Methods for Solving of the Aircraft Landing Problem. II. Approximate Solution Methods // Automation and Remote Control. – 2019. – Vol. 80. – P. 1502–1518.]
33. *Soares, I. B., De Hauwere, Y.M., Januarius, K.*, et al. Departure Management with a Reinforcement Learning Approach: Respecting CFMU Slots // IEEE 18th International Conference on Intelligent Transportation Systems. – Las Palmas de Gran Canaria, Spain. 2015.
34. *Watkins, C. J. Dayan, P. Q*-learning // Machine Learning. – 1992. – Vol. 8. – P. 279–292.
35. *Brittain, M. Wei, P.* Autonomous Aircraft Sequencing and Separation with Hierarchical Deep Reinforcement Learning // Proceedings of the International Conference for Research in Air Transportation. – Barcelona, Spain. 2018.
36. *Colen, J.* NASA sector 33 application. – 2013. – URL: https://www.nasa.gov/ centers/ames/Sector33/iOS/index.html.
37. *Henry, A., Delahaye, D., Valenzuela, A.* Conflict Resolution with Time Constraints in the Terminal Maneuvering Area Using a Distributed Q-learning Algorithm // International Conference on Research in Air Transportation (ICRAT 2022). – 2022. – Tampa, USA. – Hal-03701660.
38. *Ma, J., Delahaye, D., Sbihi, M., Mongeau, M.* Integrated Optimization of Terminal Manoeuvring Area and Airport // 6th SESAR Innovation Days. – Delft, Netherlands, 2016.

Статья представлена к публикации членом редколлегии А.А. Лазаревым.

Поступила в редакцию 10.11.2022,
после доработки 19.12.2022.
Принята к публикации 20.12.2022.

Кулида Елена Львовна – канд. техн. наук,
✉ elena-kulida@yandex.ru,

Лебедев Валентин Григорьевич – д-р техн. наук,
✉ lebedev-valentin@yandex.ru,

Институт проблем управления им. В.А. Трапезникова РАН,
г. Москва.

METHODS FOR SOLVING SOME PROBLEMS OF AIR TRAFFIC PLANNING AND REGULATION. PART II: Application of Deep Reinforcement Learning

E.L. Kulida[1] and V.G. Lebedev[2]

Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia

[1]✉ elena-kulida@yandex.ru, [2]✉ lebedev-valentin@yandex.ru

Abstract. Following part I of the survey, this paper considers the problems of improving the safety and efficiency of air traffic flows. The main challenge in conflict detection and resolution by traditional optimization methods is computation time: tens and even hundreds of seconds are required. However, this is not so much for response in real situations. Deep reinforcement learning has recently become widespread due to solving high-dimensional decision problems with nonlinearity in an acceptable time. Research works on the use of deep reinforcement learning in air traffic management have appeared in the last few years. Part II focuses on the application of this promising approach to the following problems: detecting and resolving aircraft conflicts, reducing the complexity of air traffic at the national or continental level (a large-scale problem), and increasing the efficiency of airport runways through the improved planning of aircraft landings.

Keywords: air traffic management, strategic planning of 4D trajectories, aircraft conflict detection and resolution, reinforcement learning.