УДК 519.854.2 DOI: http://doi.org/10.25728/pu.2024.1.3

ИССЛЕДОВАНИЕ ЭФФЕКТИВНОСТИ КОМБИНИРОВАННОГО ИЕРАРХИЧЕСКОГО ОПЕРАТОРА СКРЕЩИВАНИЯ В ГЕНЕТИЧЕСКОМ АЛГОРИТМЕ РЕШЕНИЯ ЗАДАЧИ ДОСТАВКИ ПОСЛЕДНЕЙ МИЛИ

В. А. Соседов

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

✉ vladyslav.sosedov@gmail.com

Аннотация. Рассматривается задача планирования маршрутов группы беспилотных летательных аппаратов в составе перспективной системы доставки последней мили, формализованная в виде двухкритериальной NP-трудной задачи многих коммивояжеров с одним депо. Применение стандартных методов оптимизации для получения точного решения неэффективно с точки зрения временны́х затрат на их реализацию, и в условиях реальной системы становится необходимым применение эвристических алгоритмов поиска приближенного решения. Для решения поставленной задачи был применен элитарный генетический алгоритм недоминирующей сортировки NSGA-II, хорошо зарекомендовавший себя в случае многокритериальной оптимизации. Для исследования эффективности применения комбинированного иерархического оператора скрещивания в сравнении со стандартными операторами скрещивания было реализовано программное средство имитационного моделирования и был проведен сравнительный анализ результатов применения различных операторов скрещивания в составе генетического алгоритма.

Ключевые слова: доставка последней мили, задача многих коммивояжеров, многокритериальная оптимизация, генетический алгоритм, оператор скрещивания.

ВВЕДЕНИЕ

Быстрая и экономичная коммерческая доставка малогабаритных товаров, заказываемых онлайн, является сложной комплексной логистической задачей, которую в настоящее время пытаются решить многие компании. Одной из наиболее перспективных технологий для решения этой задачи на заключительном этапе цепочки поставок, так называемой доставки «последней мили», является применение беспилотных летательных аппаратов (БПЛА) в составе системы доставки товаров из распределительных центров до покупателей [1]. Использование БПЛА для доставки малогабаритных грузов в городской среде эффективнее классической курьерской доставки с экономической точки зрения, так как БПЛА не ограничены отдельным статичным набором дорог и могут гибко перемещаться в трех измерениях, что существенно сокращает время и, как следствие, стоимость доставки [2]. Кроме того, автоматизация подобной системы позволяет существенно снизить число работников, задействованных в процессе транспортировки заказов, что также ведет к снижению ее общей стоимости.

Хотя доставка последней мили с использованием БПЛА и предполагает более эффективную альтернативу курьерской доставке, ее реализация в составе реальной системы ограничена техническими возможностями существующего в настоящее время аппаратного и программного обеспечения. Также стоит отметить, что реальные системы

доставки грузов предполагают наличие большого числа клиентов, которых необходимо обслужить единовременно.

В обзоре [3] показано, что задача предварительного централизованного планирования маршрутов группы БПЛА с учетом того, что каждый БПЛА может обслужить за вылет одного или нескольких клиентов, в простейшей постановке может быть сведена к *NP*-трудной задаче многих коммивояжеров (англ. *Multiple Travelling Salesman Problem*, MTSP). Для уменьшения размерности задачи оптимизации в настоящей статье предлагается разделение всей системы доставки на отдельные зоны обслуживания с единственным распределительным центром внутри и обособленное решение задач многих коммивояжеров с одним депо (англ. *Single-Depot Multiple Travelling Salesman Problem*, SD-MTSP). В § 1 показана необходимость одновременной оптимизации двух конфликтующих целевых функций и приведена формальная постановка задачи.

Применение стандартных методов оптимизации для получения точного решения приводит к экспоненциальному росту времени вычислений с ростом числа оптимизируемых параметров, и в условиях реальной системы становится необходимым применение эвристических алгоритмов поиска приближенных решений. В настоящее время основным подходом к решению этого класса задач является применение различных метаэвристических алгоритмов [4]. Для решения поставленной задачи был выбран элитарный генетический алгоритм недоминирующей сортировки (англ. *Non-dominated Sorting Genetic Algorithm II*, NSGA-II). В § 2 подробно описаны сам алгоритм NSGA-II и алгоритмы используемых генетических операторов селекции, скрещивания и мутации.

Эффективность работы генетического алгоритма в наибольшей степени зависит от возможностей локального поиска используемого оператора скрещивания (кроссовера). В § 3 приведен сравнительный анализ результатов работы алгоритма с использованием комбинированного иерархического кроссовера и таких стандартных операторов скрещивания, как частично соответствующий, циклический и порядковый кроссоверы.

1. ПОСТАНОВКА ЗАДАЧИ

Задача многих коммивояжеров с одним депо в общем виде формулируется следующим образом. Существует набор из $n > 1$ городов (точек) и m идентичных коммивояжеров. Каждый коммивояжер отправляется из одной и той же стартовой точки с номером 0, совершает тур и возвращается в исходную стартовую точку. Каждая точка, не считая стартовой, должна быть посещена ровно один раз. Решение задачи заключается в минимизации некоторой глобальной целевой функции F [5].

В стандартной постановке задачи в качестве целевой функции принимается суммарная протяженность маршрутов всех коммивояжеров. Однако минимизация такой целевой функции в отсутствие дополнительных ограничений на маршруты коммивояжеров приводит к сильной несбалансированности маршрутов в получаемых решениях (рис. 1).

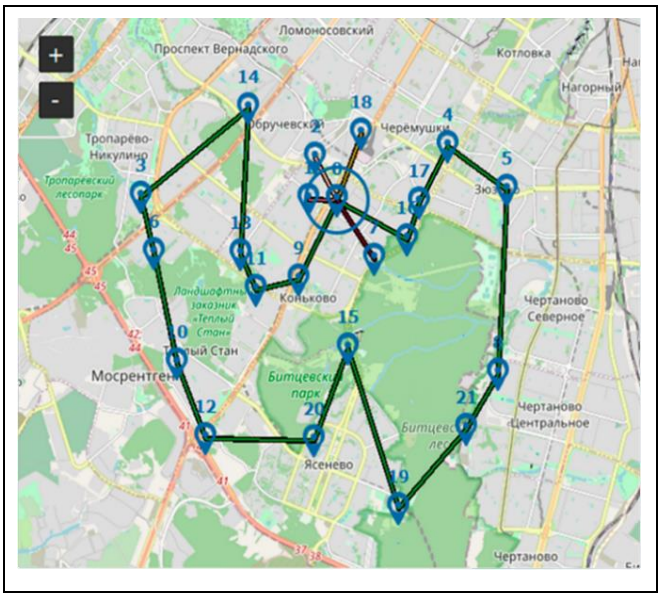

Рис. 1. Типичный результат решения задачи в стандартной постановке

С практической точки зрения использование нескольких коммивояжеров направлено на сокращение времени обслуживания всех клиентов, поэтому для обеспечения сбалансированности маршрутов коммивояжеров по их протяженности используются различные вариации минимаксной постановки задачи, где в качестве целевой функции может использоваться, например, протяженность максимального маршрута или так называемая «степень несбалансированности», которая рассчитывается как разность протяженностей максимального и минимального маршрутов. В свою очередь, такой подход может привести к тому, что маршруты коммивояжеров будут нерациональными в смысле минимизации их протяженностей

(рис. 2) и время обслуживания всех клиентов также будет неоптимальным.

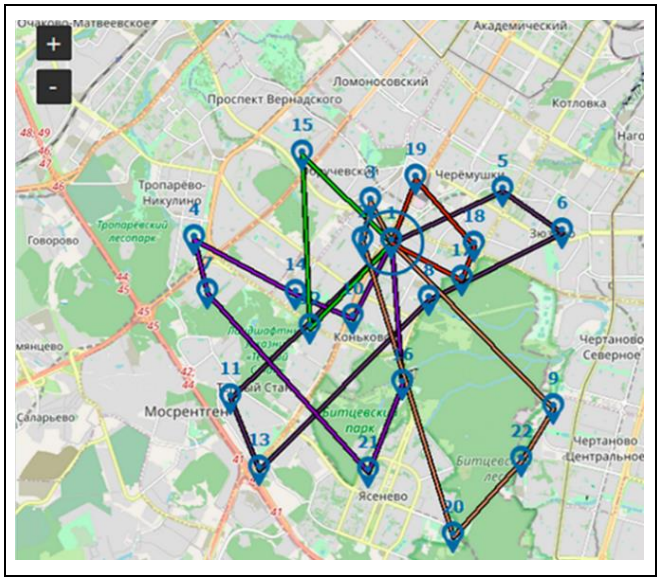

Рис. 2. Типичный результат решения задачи в минимаксной постановке

Таким образом, минимизация суммарной протяженности и обеспечение сбалансированности маршрутов – это две конфликтующие задачи оптимизации, которые необходимо рассматривать совместно. Многокритериальный подход к решению задачи многих коммивояжеров зарекомендовал себя эффективным [6–8], поскольку он позволяет сглаживать недостатки конфликтующих целевых функций без введения дополнительных ограничений на решения. Кроме того, наличие нескольких целевых функций в задаче закономерно приводит к появлению набора Парето-оптимальных решений вместо единственного оптимального решения, что позволяет гибко выбирать лучшее в некотором смысле решение при наличии дополнительной информации о задаче.

Двухкритериальная задача многих коммивояжеров с одним депо может быть формализована [6] с использованием ориентированного графа $G = (V, A)$, где V – множество вершин и A – множество дуг. С графом связана симметричная весовая матрица (матрица расстояний) $C = (c_{ij})$, $(i, j) \in A$, которая может быть определена с учетом ограничений, налагаемых внешней средой функционирования реальной системы доставки. В настоящей статье рассматривается простейший вариант определения значений c_{ij} евклидовыми расстояниями между точками. Пусть x_{ijk} – двоичная переменная, принимающая значение 1, если k-й коммивояжер проходит по дуге графа (i, j), и 0 – в противном случае; u_i – количество точек, посещенных коммивояжером на пути от стартовой точки с номером 0 до точки с номером i. С учетом введенных обозначений формальная запись задачи имеет вид:

$$F = \left(f_1,\ f_2 \right) \to \min, \tag{1}$$

$$f_1 = \sum_{k=1}^{m} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ijk}, \tag{2}$$

$$f_2 = \max_{1 \le k \le m} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ijk} - \min_{1 \le k \le m} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ijk}, \tag{3}$$

где $x_{ijk} \in \{0, 1\} \quad \forall (i, j) \in A, \ k = 1, \ldots, m,$ такие, что

$$\sum_{j=1}^{n} x_{0jk} = 1, \ k = 1, \ldots, m, \tag{4}$$

$$\sum_{i=1}^{n} x_{i0k} = 1, \ k = 1, \ldots, m, \tag{5}$$

$$\sum_{k=1}^{m} \sum_{i=1}^{n} x_{ijk} = 1, \ j = 1 \ldots n, \ i \ne j, \tag{6}$$

$$\sum_{k=1}^{m} \sum_{j=1}^{n} x_{ijk} = 1, \ i = 1 \ldots n, \ i \ne j, \tag{7}$$

$$\sum_{i=1}^{n} x_{ijk} = \sum_{i=1}^{n} x_{jik}, \ j = 1, \ldots, n, \ k = 1, \ldots, m, \ i \ne j, \tag{8}$$

$$u_i - u_j + (n - m) \cdot \sum_{k=1}^{m} x_{ijk} \le n - m - 1, \quad 2 \le i \ne j \le n. \tag{9}$$

Выражения (2) и (3) определяют две целевые функции, которые необходимо минимизировать совместно (1): суммарную протяженность маршрутов всех коммивояжеров и разность протяженностей максимального и минимального маршрутов соответственно. Условия (4) и (5) гарантируют, что ровно m коммивояжеров покинут стартовую точку и возвратятся в нее. Условия (6)–(8) гарантируют, что каждая точка, не считая стартовой, будет посещена ровно один раз. Ограничение (9) представляет собой классическую формулировку ограничения исключения промежуточных туров (англ. *subtour elimination constraint*, SEC), которое гарантирует, что решение не будет содержать маршрутов коммивояжеров, не включающих в себя уже посещенные точки.

2. ОПИСАНИЕ АЛГОРИТМА

2.1. Представление хромосомы

Ключом к получению хороших решений задачи оптимизации с применением генетического алгоритма является корректное представление особи или хромосомы, которая должна точно определять решение задачи и позволять генетическим операторам эффективно генерировать лучшие решения по мере продолжения итеративного эволюционного процесса. В работе [9] был предложен пригодный для решения задачи многих коммивояжеров метод представления хромосомы, состоящей из двух частей, позволяющий уменьшить избыточность пространства решений (в сравнении с однострочным и двухстрочным методами представления хромосомы) и, как следствие, повысить эффективность работы алгоритма.

Хромосома состоит из двух частей (рис. 3): первая часть содержит набор из $n - 1$ чисел, характеризующих порядок посещения точек в маршрутах коммивояжеров, вторая часть состоит из $m - 1$ разделителей, которые делят первую часть на m групп. Каждая группа представляет собой тур одного из коммивояжеров (рис. 4).

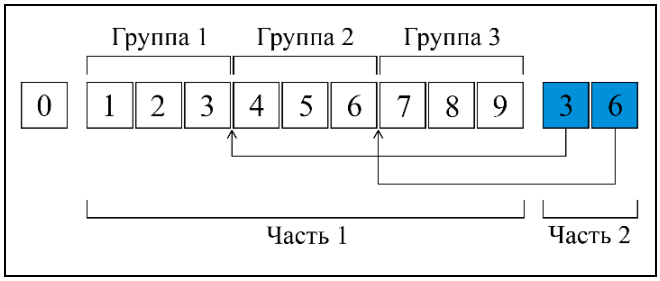

Рис. 3. Представление хромосомы

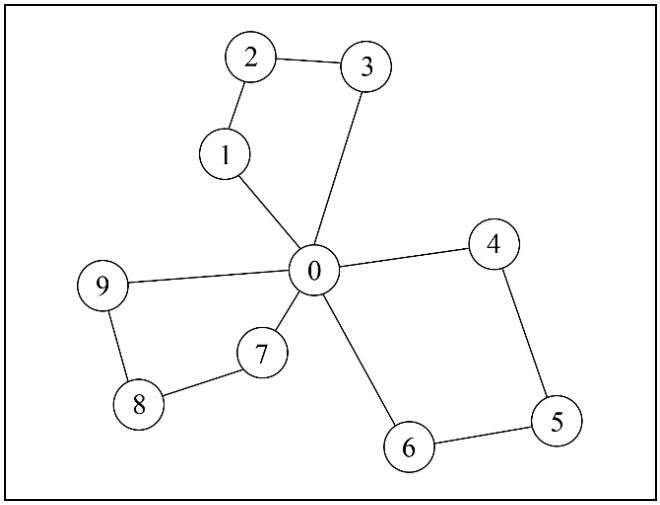

Рис. 4. Визуализация решения задачи

2.2. Операторы скрещивания

Известно, что используемый в генетическом алгоритме оператор скрещивания (кроссовер) оказывает наибольшее влияние на возможности локального поиска и эффективность работы алгоритма в целом. Наиболее широкое распространение в работах, посвященных решению комбинаторных задач оптимизации, получили такие операторы скрещивания, как частично соответствующий кроссовер (англ. *Partially-Mapped Crossover*, PMX), порядковый кроссовер (*Order Crossover*) и циклический кроссовер (*Cycle Crossover*). Принципы их функционирования были описаны в множестве работ (см., например, книги [10, 11]). Используемые в настоящей работе алгоритмы стандартных операторов скрещивания были адаптированы для соответствия методу представления хромосомы, состоящей из двух частей. Модифицированные алгоритмы операторов скрещивания подробно описаны далее.

Алгоритм работы частично соответствующего кроссовера таков (рис. 5).

Шаг 1. На вход алгоритма подается пара особей-родителей. Из первых частей родительских хромосом *PA* и *PB* случайным образом равномерно выбираются две точки разреза. Последовательности генов между двумя точками разреза называются секциями отображения.

Шаг 2. Секции отображения меняются местами и копируются с соблюдением позиций генов в первые части хромосом особей-потомков *CA* и *CB*.

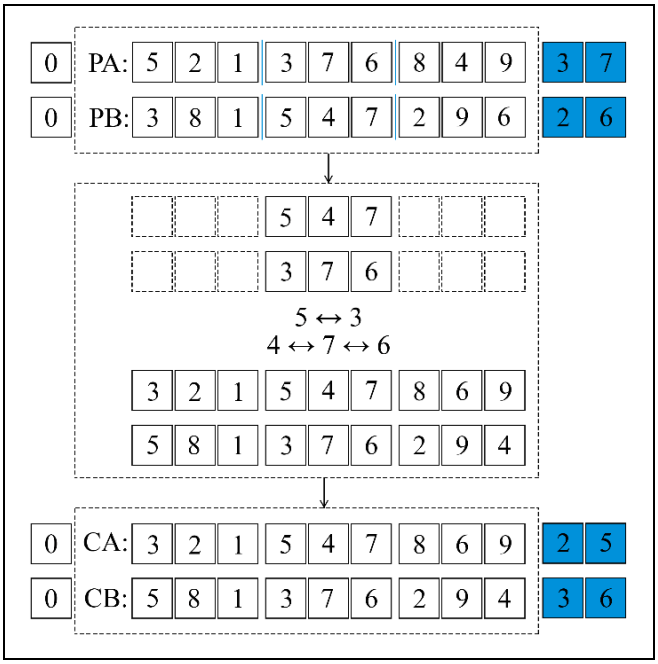

Рис. 5. Модифицированный алгоритм работы частично соответствующего кроссовера

Прочие гены при этом считаются неопределенными.

Шаг 3. Определяется взаимосвязь генов между двумя секциями отображения.

Шаг 4. Неопределенные в *CA* и *CB* гены заполняются путем копирования генов соответствующего родителя. Если ген уже присутствует в особи-потомке, он заменяется в соответствии с его отображением.

Шаг 5. Вторые части хромосом особей-потомков генерируются случайным образом.

Алгоритм работы порядкового кроссовера таков (рис. 6).

Шаг 1. На вход алгоритма подается пара особей-родителей. Из первых частей родительских хромосом *PA* и *PB* случайным образом выбираются две точки разреза.

Шаг 2. Полученные последовательности генов меняются местами и копируются с соблюдением позиций генов в первые части хромосом особей-потомков *CA* и *CB*. Прочие гены при этом считаются неопределенными.

Шаг 3. Определяются порядки следования генов слева направо в *PA* и *PB*.

Шаг 4. Неопределенные в *CA* и *CB* гены заполняются в соответствии с порядком следования генов соответствующих родителей, начиная со второй точки разреза. Если ген уже присутствует в особи-потомке, он пропускается.

Шаг 5. Вторые части хромосом особей-потомков генерируются случайным образом.

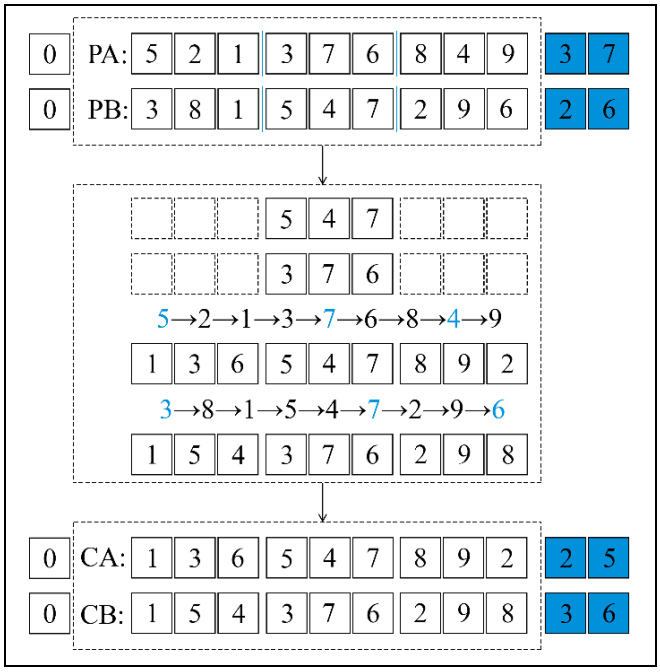

Рис. 6. Модифицированный алгоритм работы порядкового кроссовера

Алгоритм работы циклического кроссовера таков (рис. 7).

Шаг 1. На вход алгоритма подается пара особей-родителей. Из первой части родительской хромосомы *PA* случайным образом выбирается стартовая позиция цикла.

Шаг 2. Ген, занимающий стартовую позицию в *PA*, записывается в первую часть хромосомы потомка с соблюдением его позиции. Ген, занимающий ту же позицию в первой части родительской хромосомы *PB*, не может быть записан в хромосому потомка на эту же позицию, поэтому он записывается в хромосому потомка в соответствии с его позицией в *PA*. Цикл продолжается до нахождения в *PB* гена, занимавшего стартовую позицию в *PA*.

Шаг 3. После завершения цикла оставшиеся неопределенные гены копируются из *PB* с соблюдением их позиций.

Шаг 4. Вторая часть хромосомы особи-потомка генерируется случайным образом.

Шаг 5. Для формирования второй особи-потомка особи-родители меняются местами и *шаги 1-4* алгоритма повторяются.

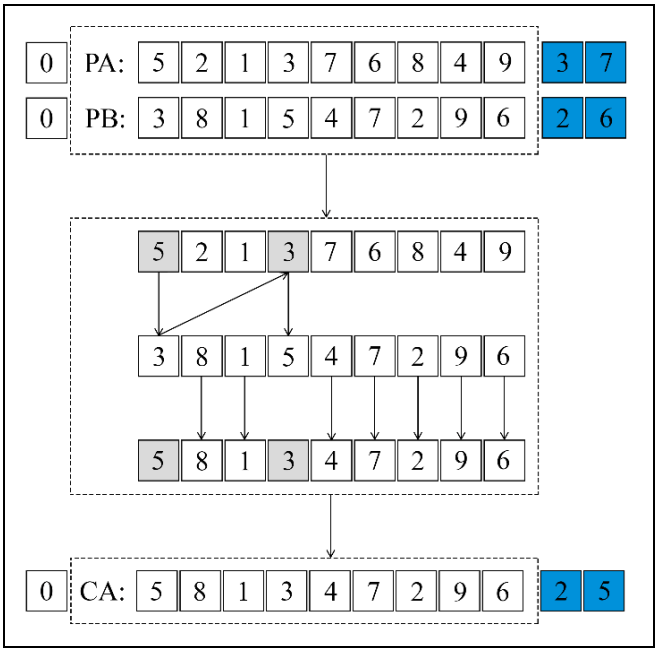

Рис. 7. Модифицированный алгоритм работы циклического кроссовера

Описанные операторы скрещивания являются универсальными для комбинаторных задач оптимизации и позволяют генерировать лучшие решения по мере продолжения итеративного эволюционного процесса генетического алгоритма, однако в практических приложениях их возможности локального поиска сравнительно невелики.

В работе [12] для решения задачи многих коммивояжеров был предложен комбинированный иерархический оператор скрещивания (*Combined HGA*), обеспечивающий высокую эффективность локального поиска путем учета расстояний между точками в основном процессе формирования особи-потомка и хорошее разнообразие популяции благодаря применению разных подходов к формированию двух особей-потомков.

Основной алгоритм комбинированного иерархического оператора скрещивания таков.

Шаг 1. На вход алгоритма подается пара хромосом *PA* и *PB*. Из *PA* случайным образом выбирается ген PA_k, который записывается в начало хромосомы потомка.

Шаг 2. В зависимости от заданного направления поиска из хромосом родителей выбираются два последующих PA_{k+1}, PB_{k+1} или два предыдущих PA_{k-1}, PB_{k-1} гена.

Шаг 3. Из хромосом родителей удаляется ген PA_k.

Шаг 4. Сравнивается пара расстояний c_{ij} между выбранными на втором шаге генами и PA_k. Ген, имеющий меньшее расстояние, записывается в особь-потомка и становится новым геном PA_k. Алгоритм продолжает работу, начиная с *шага 2*, пока длина хромосомы родителя $PA > 1$.

Авторами работы [12] было показано, что получаемые с помощью такого алгоритма особи-потомки будут иметь разные показатели по двум целевым функциям в зависимости от представления входных данных *PA* и *PB*. Для получения хорошо сбалансированного набора решений было предложено два метода формирования входных данных.

Метод формирования первой особи-потомка (рис. 8) заключается в том, что в качестве входных данных основного алгоритма выбираются первые части хромосом родителей. Результатом работы алгоритма будет первая часть хромосомы потомка, вторая часть хромосомы потомка выбирается случайным образом из вторых частей хромосом родителей. Большинство получаемых таким образом

Рис. 8. Алгоритм формирования первой особи-потомка

потомков будут иметь достаточно сбалансированные маршруты коммивояжеров, но не будут способствовать уменьшению общей протяженности всех маршрутов.

Метод формирования второй особи-потомка (рис. 9) состоит в предварительном декодировании хромосом родителей в однострочное представление: в начало первой части хромосомы и на определяемые разделителями позиции добавляются гены стартовой точки 0, вторая часть хромосомы удаляется. Декодированные родительские хромосомы подаются на вход основного алгоритма.

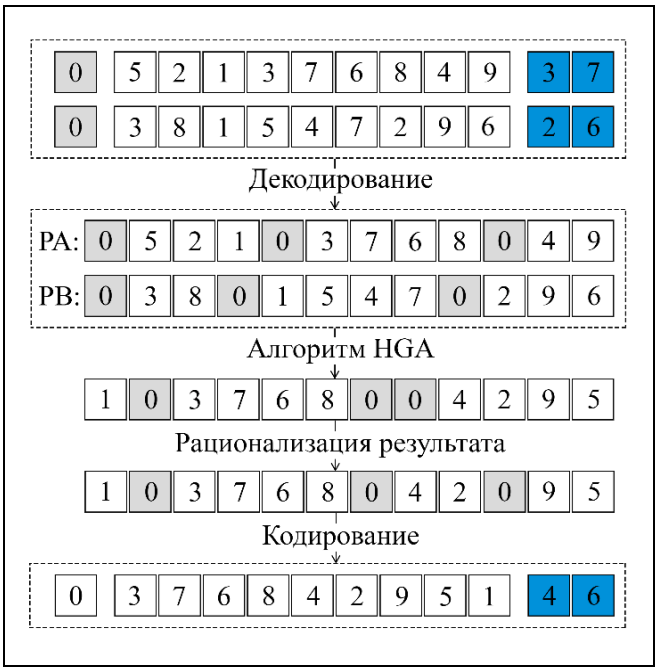

Рис. 9. Алгоритм формирования второй особи-потомка

Результат работы алгоритма подвергается процедуре рационализации, состоящей в удалении «нулевых» маршрутов (перенос правых смежных генов стартовой точки 0 в случайно определяемую позицию правее по хромосоме), а затем кодируется в изначальное представление хромосомы, состоящей из двух частей. Большинство получаемых таким образом потомков будут способствовать уменьшению общей протяженности всех маршрутов, однако не будут способствовать уменьшению степени несбалансированности маршрутов.

2.3. Операторы селекции и мутации

Оператор селекции осуществляет отбор особей, обладающих наибольшим значением функции приспособленности, из которых формируется набор допущенных к скрещиванию особей. В предложенном алгоритме используется стандарт-

ный для генетического алгоритма недоминирующей сортировки оператор бинарной турнирной селекции.

Операторы мутации предотвращают сужение области поиска оптимального решения из-за попадания в локальный минимум путем редактирования генов у создаваемых потомков. В предложенном алгоритме были использованы четыре различных оператора мутации [10, 13]: мутация вставки (*Insertion Mutation*), мутация взаимообмена (*Exchange Mutation*), мутация инверсии (*Inversion Mutation*) и мутация скремблирования (*Scramble Mutation*). Все операторы мутации имеют равную вероятность вызова.

2.4. Генетический алгоритм недоминирующей сортировки NSGA-II

Для решения поставленной задачи в качестве метаэвристического алгоритма оптимизации был применен предложенный в работе [14] элитарный генетический алгоритм недоминирующей сортировки NSGA-II (рис. 10), хорошо зарекомендовавший себя в случае многокритериальной оптимизации. Алгоритм основан на процедуре быстрой недоминирующей сортировки множества решений на фронты (англ. *fast non-dominated sort*, FNDS), которая обеспечивает высокую скорость сходимости алгоритма, и на процедуре сортировки по степени скученности решений в пространстве функционалов (*crowding-distance sort*), которая обеспечивает хорошее разнообразие формируемой популяции.

Быстрая недоминирующая сортировка состоит в определении ранга r для каждой особи и объединении особей с одинаковым рангом в подмножества $\mathcal{F}_r$ – фронты ранга r.

Определение. Решение p называется доминирующим над решением q, если одновременно выполняются два условия:

1. Решение p не хуже решения q по всем функционалам.

2. Решение p строго лучше решения q хотя бы по одному функционалу. ♦

Таким образом, для поставленной задачи двухкритериальной оптимизации можно записать условие доминирования особи p над особью q:

$$\begin{gathered}\big(f_1(p)\le f_1(q)\wedge f_2(p)\le f_2(q)\big)\wedge\\ \wedge\big(f_1(p)< f_1(q)\vee f_2(p)< f_2(q)\big).\end{gathered}\qquad(10)$$

Представим условие (10) как $p \prec q$ для удобства дальнейшего изложения.

Алгоритм быстрой недоминирующей сортировки таков.

Шаг 1. Для каждой особи p в популяции выполняются следующие действия:

- Инициируется список $S_p = \emptyset$ особей популяции, над которыми доминирует особь p.
- Инициируется счетчик $n_p = 0$ особей популяции, которые доминируют над особью p.

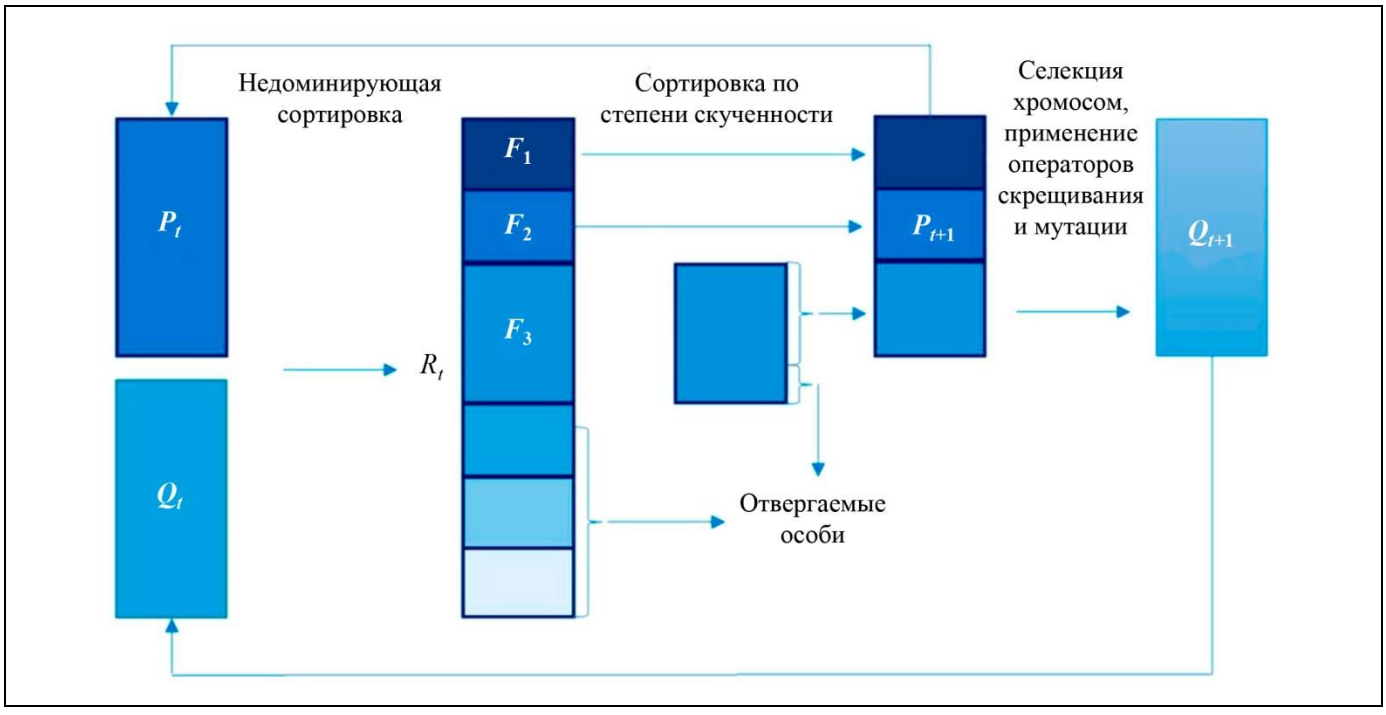

Рис. 10. Процесс оптимизации популяции в алгоритме NSGA-II

• Для каждой особи q в популяции проверяется условие доминирования (10):

○ если $p \prec q$, особь q добавляется в список S_p, т. е. $S_p = S_p \cup \{q\}$.

○ если $q \prec p$, счетчик n_p увеличивается на 1, т. е. $n_p = n_p + 1$.

• Если для особи p счетчик $n_p = 0$, она записывается в первый недоминирующий фронт $\mathcal{F}_1 \cup \{p\}$.

Шаг 2. Инициируется счетчик фронтов $i = 1$.

Шаг 3. Пока i-й фронт $\mathcal{F}_i \neq \emptyset$, выполняются следующие действия:

• Инициируется список $Q = \emptyset$ для хранения особей $(i + 1)$-го фронта.

• Для каждой особи $p \in \mathcal{F}_i$ рассматривается список S_p: если для особи $q \in S_p$ выполняется равенство $n_q - 1 = 0$, она записывается в список $Q = Q \cup \{q\}$.

• Счетчик фронтов увеличивается на 1, т. е. $i = i + 1$.

• Формируется i-й фронт $\mathcal{F}_i = Q$.

Процедура продолжается до идентификации всех фронтов $\mathcal{F}_r$, $r = 1,\ldots, r_{\max}$.

Сортировка по степени скученности решений в пространстве функционалов необходима для отбора решений в пределах одного фронта. Для каждой особи $\mathcal{F}_r(i) \in \mathcal{F}_r$ вычисляется величина расстояния

$$\operatorname{dist}\left(\mathcal{F}_r(i)\right) = \sum_{m=1}^{2} \operatorname{dist}_m\left(\mathcal{F}_r(i)\right),$$

где $\operatorname{dist}_m\left(\mathcal{F}_r(i)\right)$ – расстояние до m-го функционала, которое рассчитывается следующим образом.

Шаг 1. Решения в пределах фронта $\mathcal{F}_r$ длиной l сортируются по возрастанию значений m-го функционала. Тогда для граничных решений фронта справедливы выражения

$$\mathcal{F}_r(1) = f_m^{\min}, \quad \mathcal{F}_r(l) = f_m^{\max}.$$

Шаг 2. Искомым расстояниям для граничных решений присваивается максимальное значение расстояния $\operatorname{dist}_m\left(\mathcal{F}_r(1)\right) = \operatorname{dist}_m\left(\mathcal{F}_r(l)\right) = 10^9$, а расстояния промежуточных решений определяются по формуле

$$\operatorname{dist}_m\left(\mathcal{F}_r(i)\right) = \frac{f_m^{\mathcal{F}_r(i+1)} - f_m^{\mathcal{F}_r(i-1)}}{f_m^{\max} - f_m^{\min}},$$
$$i = 2,\ldots, l-1.$$

Алгоритм NSGA-II (см. рис. 10) состоит из пяти шагов.

Шаг 1. Начальная популяция родителей P_0 размером N особей формируется случайным образом. На ее основе при помощи генетических операторов селекции, скрещивания и мутации генерируется начальная популяция потомков Q_0 также размером N. Поскольку элитизм вводится путем сравнения текущей популяции с ранее найденными наилучшими решениями, процесс формирования последующих популяций будет отличаться.

Шаг 2. Популяции родителей и потомков объединяются в множество $R_t = P_t \cup Q_t$, которое разделяется на фронты F_r, $r = 1,\ldots, r_{\max}$, при помощи процедуры быстрой недоминирующей сортировки.

Шаг 3. Из множества R_t выбираются N лучших особей, которые формируют новую популяцию родителей P_{t+1}. Лучшими считаются особи с меньшим рангом r. Если очередной фронт не может быть записан целиком в P_{t+1}, он подвергается процедуре сортировки по степени скученности решений в пространстве функционалов, лучшими считаются особи с бóльшим значением расстояния скученности.

Шаг 4. На основе популяции P_{t+1} при помощи генетических операторов селекции, скрещивания и мутации генерируется новая популяция потомков Q_{t+1}.

Шаг 5. Алгоритм продолжает работу, начиная с *шага 2*, пока не будет достигнут терминальный критерий алгоритма (заданное число поколений или достаточная однородность популяции).

3. ОРГАНИЗАЦИЯ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ И ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ

Для организации вычислительных экспериментов с целью исследования влияния используемого оператора скрещивания на эффективность работы алгоритма было реализовано программное средство имитационного моделирования, которое позволяет:

• задавать состояние системы – набор точек доставки $\{n\}$, количество используемых коммивояжеров m;

• задавать параметры работы алгоритма – размер популяции N, количество итераций алгоритма N_{Gen}, количество повторений алгоритма N_{Rep}, вероятность вызова оператора скрещивания $0 \le p_x \le 1$, вероятность вызова оператора мутации $0 \le p_m \le 1$, зерно генератора псевдослучайных чисел;

• выбирать используемые в алгоритме генетические операторы скрещивания и мутации;

• в автоматическом режиме набирать статистику по поколениям и визуализировать конечный набор Парето-оптимальных решений;

• визуализировать конечное решение на интерактивной карте, если точки доставки в используе-

мом наборе имеют привязку к географическим координатам;

• создавать новые наборы данных на интерактивной карте.

Программное средство было реализовано на языке программирования Python 3.9.6, все вычисления производились на процессоре AMD Ryzen 7 5800X3D, 4.70 ГГц и 64 Гб ОЗУ.

Для всех вычислительных экспериментов размер популяции N принимался равным 100 особям, количество повторений алгоритма N_{Rep} = 1, вероятность вызова оператора скрещивания p_x = 1, вероятность вызова оператора мутации p_m = 0,05. Для проведения сравнительного анализа результатов работы алгоритма с использованием описанных операторов скрещивания в качестве наборов данных использовались эталонные тесты berlin52, eli76 и rat99 [15–17], в которых первая точка принималась за стартовую (условия приведены в таблице).

Условия вычислительных экспериментов

Набор данных	Число точек n	Число коммивояжеров m	Число итераций N_{Gen}
berlin52	52	5	1400
eli76	76	7	1800
rat99	99	7	2200

Финальные недоминированные Парето-фронты для соответствующих наборов данных, представляющие собой наборы возможных решений задачи многих коммивояжеров, изображены на графиках (рис. 11). По оси абсцисс каждого графика откладывается общая протяженность маршрутов, по оси ординат – разность между максимальной и минимальной протяженностью маршрутов (обе величины безразмерные).

Парето-фронты, полученные при помощи алгоритма с использованием комбинированного иерархического кроссовера, во всех эталонных тестах показывают лучшие результаты с точки зрения совместной минимизации двух целевых функций по сравнению со стандартными операторами скрещивания.

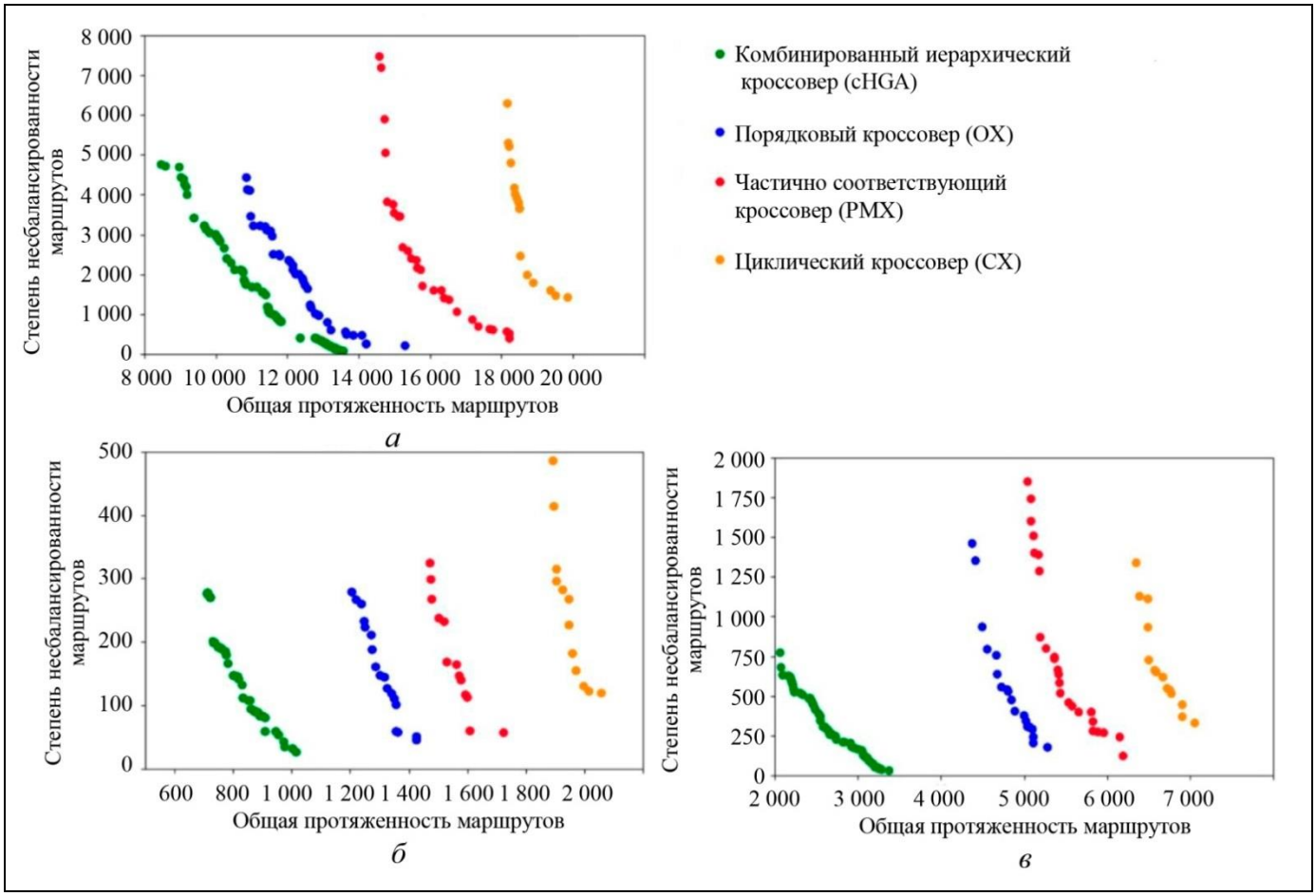

Рис. 11. Сравнительный анализ финальных недоминированных Парето-фронтов для наборов данных: *а* - berlin52, m=5; *б* – eli76, m=7; *в* – rat99, m=7

Для оценки скорости сходимости алгоритма с использованием комбинированного иерархического оператора скрещивания были построены графики эволюции минимальных значений двух целевых функций, соответствующие крайним особям текущего недоминированного Парето-фронта. Для всех эталонных наборов данных прослеживается следующая закономерность: на первых поколениях значения целевых функций уменьшаются стремительно, затем скорость их изменения резко снижается (рис. 12). Таким образом, в составе реальной системы доставки последней мили для экономии вычислительных и, как следствие, временны́х ресурсов можно использовать наборы решений, полученные за малое количество итераций предложенного алгоритма.

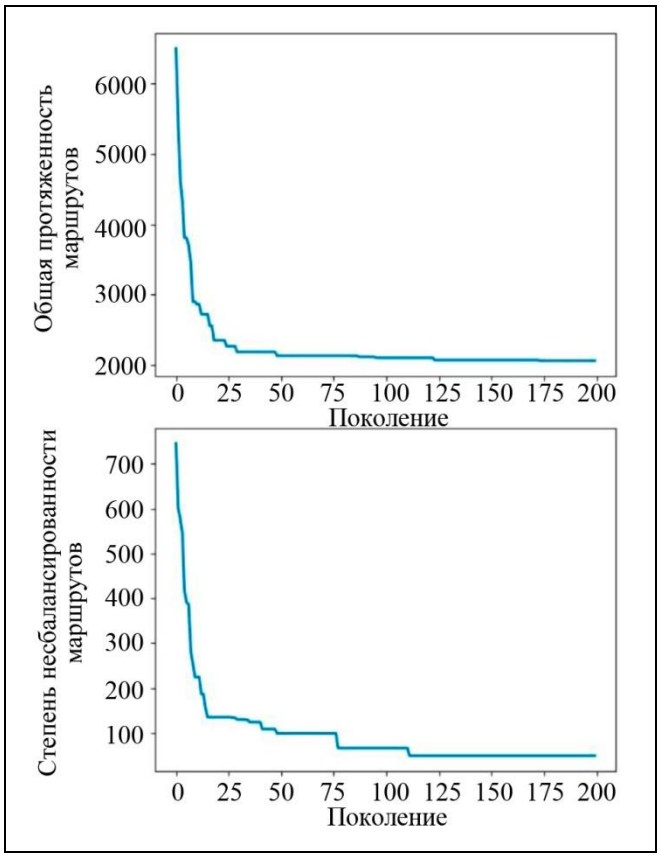

Рис. 12. Эволюция значений целевых функций для набора данных rat99, *m*=7

Так, например, для набора данных Moscow-ICS, созданного при помощи интерактивной карты программного средства, было получено решение (рис. 13) при следующих условиях работы алгоритма: $n = 41$, $m = 5$, $N_{Gen} = 200$. Для визуализации решения задачи (рис. 14) из полученного набора Парето-оптимальных решений было выбрано Парето-эффективное решение, занимающее серединное положение в финальном фронте.

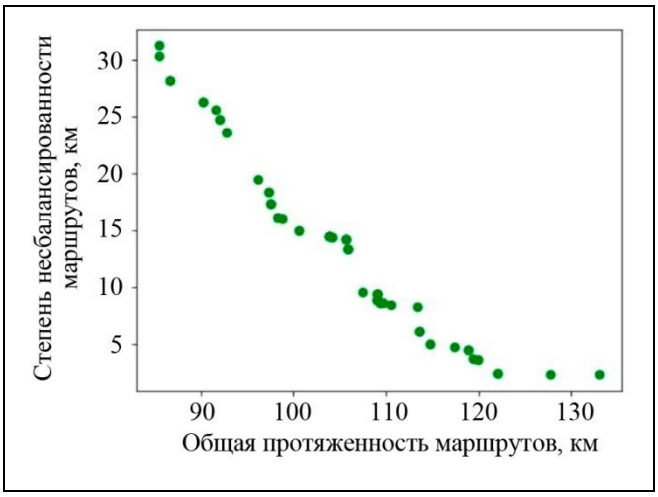

Рис. 13. Финальный недоминированный Парето-фронт для набора данных Moscow-ICS

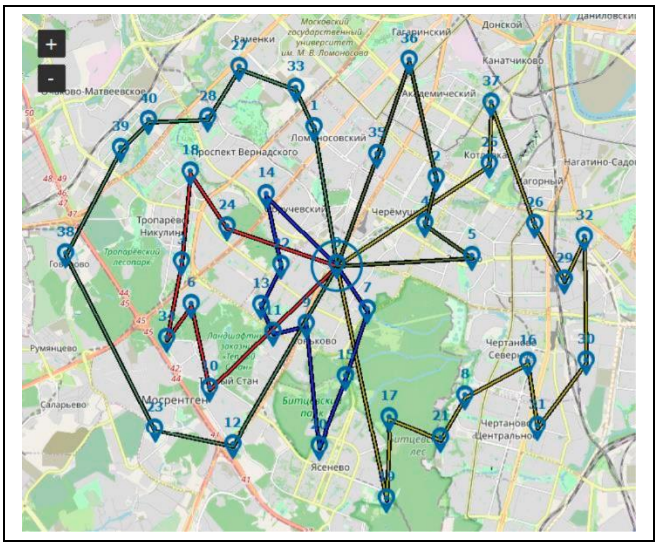

Рис. 14. Визуализация решения задачи для набора данных Moscow-ICS

ЗАКЛЮЧЕНИЕ

Рассмотрена задача планирования маршрутов беспилотных летательных аппаратов в составе перспективной системы доставки последней мили в виде двухкритериальной задачи многих коммивояжеров с одним депо. Показана необходимость совместной оптимизации двух конфликтующих целевых функций. Описаны использованные генетические операторы. Реализовано программное средство для решения поставленной задачи, в основу которого положен элитарный генетический алгоритм недоминирующей сортировки NSGA-II.

Проведены вычислительные эксперименты и сравнительный анализ эффективности применения различных операторов скрещивания в составе ге-

нетического алгоритма решения двухкритериальной задачи многих коммивояжеров. Показано, что комбинированный иерархический кроссовер обеспечивает наилучшие результаты с точки зрения совместной оптимизации двух целевых функций. Исследование скорости сходимости алгоритма с использованием комбинированного иерархического кроссовера показало, что предложенный алгоритм позволяет получать приемлемые решения в кратчайшие сроки.

ЛИТЕРАТУРА

1. *Baur, S.* Cargo drones: A potential gamechanger in the logistics industry // Roland Berger. – 2022. – URL: https://www.rolandberger.com/en/Insights/Publications/Cargo-drones-A-potential-gamechanger-in-the-logistics-industry.html (дата обращения: 23.09.2023). [Accessed September 23, 2023.]
2. *Moadab, A., Farajzadeh, F., Fatahi Valilai, O.* Drone routing problem model for last-mile delivery using the public transportation capacity as moving charging stations // Scientific Reports. – 2022. – Vol. 12, no. 1. – P. 1–16.
3. *Khoufi, I., Laouiti, A., Adjih, C.* A Survey of Recent Extended Variants of the Traveling Salesman and Vehicle Routing Problems for Unmanned Aerial Vehicles // Drones. – 2019. – Vol. 3, no. 3. – Art. no. 66.
4. *Германчук М.С., Лемтюжникова Д.В., Лукьяненко В.А.* Метаэвристические алгоритмы для задач многоагентных задач маршрутизации // Проблемы управления. – 2020. – № 6. – С. 3–13. [*Germanchuk, M.S., Lemtyuzhnikova, D.V., Lukianenko, V.A.* Metaheuristic Algorithms for Multi-Agent Routing Problems // Control Sciences. – 2020. – No. 6. – P. 3–13. (In Russian)]
5. *Bektas, T.* The multiple traveling salesman problem: an overview of formulations and solution procedures // Omega. – 2006. – Vol. 34, no. 3. – P. 209–219.
6. *Necula, R., Breaban, M., Raschip, M.* Tackling the bi-criteria facet of multiple traveling salesman problem with ant colony systems // IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI). – Vietri sul Mare, 2015. – P. 873–880.
7. *Bolanos, R., Echeverry, M., Escobar, J.* A multiobjective non-dominated sorting genetic algorithm (NSGA-II) for the Multiple Travelling Salesman Problem // Decision Science Letters. – 2015. – Vol. 4. – P. 559–568.
8. *Alves, R.M.F., Lopes, C.R.* Using Genetic Algorithms to minimize the distance and balance the routes for the multiple Travelling Salesman Problem // IEEE Congress on Evolutionary Computation (CEC). – Sendai, 2015. – P. 3171–3178.
9. *Carter, A.E., Ragsdale, C.* A new approach to solving the multiple traveling salesperson problem using genetic algorithms // European Journal of Operational Research. – 2005. – Vol. 175, no. 1. – P. 246–257.
10. *Саймон Д.* Алгоритмы эволюционной оптимизации. – М.: ДМК Пресс, 2020. – 1002 с. [Simon, D. Evolutionary Optimization Algorithms. – New York: John Wiley & Sons, 2013. – 784 p.]
11. *Гладков Л.А., Курейчик В.В., Курейчик В.М.* Генетические алгоритмы. 2-е изд., испр. и доп. – М.: ФИЗМАТЛИТ, 2010. – 368 с. [Gladkov L.A., Kureichik V.V., Kureichik V.M. Geneticheskie algoritmy. 2-e izd., ispr. i dop. – M.: FIZMATLIT, 2010. – 368 s. (In Russian)]
12. *Shuaia, Y., Yunfengaand, S., Kai, Z.* An effective method for solving multiple travelling salesman problem based on NSGA-II // Systems Science & Control Engineering. – 2019. – Vol. 7, no. 2. – P. 108–116.
13. *Soni, N., Kumar, T.* Study of Various Mutation Operators in Genetic Algorithms // International Journal of Computer Science and Information Technologies. – 2014. – Vol. 5, no. 3. – P. 4519–4521.
14. *Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.* A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II // IEEE Transactions on Evolutionary Computation. – 2002. – Vol. 6, no. 2. – P. 182–197.
15. *Benchmark* data for the Single-Depot Multiple Traveling Salesman Problem (multiple-TSP). – Iaşi: Alexandru Ioan Cuza University (UAIC). – URL: https://profs.info.uaic.ro/~mtsplib/ (дата обращения: 23.09.2023).
16. *TSPLIB. Symmetric Traveling Salesman Problem (TSP).* – Heidelberg: University of Heidelberg. – URL: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/ (дата обращения: 23.09.2023). [Accessed September 23, 2023.]
17. TSPLIB. – Capacitated Vehicle Routing Problem (CVRP). Heidelberg: University of Heidelberg. – URL: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/vpr (дата обращения: 23.09.2023). [Accessed September 23, 2023.]

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

Поступила в редакцию 15.05.2023,
после доработки 12.11.2023.
Принята к публикации 29.11.2023.

Соседов Владислав Александрович – инженер, Институт проблем управления им. В.А. Трапезникова РАН, г. Москва,
✉ vladyslav.sosedov@gmail.com
ORCID iD: https://orcid.org/0009-0001-0920-2579

COMBINED HIERARCHICAL CROSSOVER IN A GENETIC ALGORITHM FOR LAST-MILE DELIVERY: EFFICIENCY ANALYSIS

V. A. Sosedov

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

✉ vladyslav.sosedov@gmail.com

Abstract. This paper considers routing for a group of unmanned aerial vehicles within a promising last-mile delivery system. The routing problem is reduced to the bi-criteria single-depot multiple traveling salesman problem and formalized using a directed graph. Being NP-hard, this problem cannot be efficiently solved by standard exact optimization methods. Therefore, heuristic algorithms should be applied to obtain good approximate solutions in a short time. The problem is solved using NSGA-II, the widespread elitist non-dominated sorting genetic algorithm that demonstrates good results in multicriteria optimization. Some chromosome representation and crossing and mutation operators are implemented in the algorithm. A simulation software tool is presented to investigate the influence of the crossing operators used on the convergence speed of the algorithm. Finally, several genetic crossing operators (Partially-Mapped Crossover, Order Crossover, Cycle Crossover, and Combined Hierarchical Crossover) are compared in terms of efficiency.

Keywords: last-mile delivery, multiple traveling salesman problem, multicriteria optimization, genetic algorithm, crossover.