Математические структуры и моделирование 2019. №4(52). С. 33–51

УДК 519.8
DOI 10.24147/2222-8772.2019.4.33-51

ПРИМЕНЕНИЕ ПРИНЦИПОВ МЕМЕТИКИ К РЕШЕНИЮ ЗАДАЧИ КОММИВОЯЖЁРА

А.В. Еремеев[1]
д.ф.-м.н., доцент, e-mail: eremeev@ofim.oscsbras.ru
Ю.В. Коваленко[1,2]
к.ф.-м.н., e-mail: julia.kovalenko.ya@yandex.ru

[1]Институт научной информации по общественным наукам РАН, Москва, Россия

[2]Омский государственный университет им. Ф.М. Достоевского, Омск, Россия

Аннотация. Рассматривается популяционный алгоритм для несимметричной задачи коммивояжёра, основанный на принципах меметики. Начальная популяция пробных решений строится с помощью жадных конструктивных эвристик. Все особи начальной и финальной популяций улучшаются посредством эвристики локального поиска, основанной на окрестности 3-opt. Баланс между интенсивностью использования локальной оптимизации и интенсивностью использования популяционного поиска регулируется с помощью адаптивного перезапуска алгоритма. Приводятся результаты численных экспериментов на известных тестовых примерах, показывающие перспективность предложенного алгоритма.

Ключевые слова: мемы, генетический алгоритм, локальная оптимизация, вычислительный эксперимент.

Введение

В настоящее время при решении задач оптимизации широко применяются эволюционные методы поиска, основанные на принципах меметики. Меметика представляет собой подход к построению эволюционных моделей передачи культурной информации с использованием концепции мемов. Термин «мем» был введён Р. Докинзом в книге «Эгоистичный ген» [10]. По мнению автора, человеческая эволюция движима так называемыми репликаторами двух типов: биологическим и культурным. Репликатором первого типа является «ген» как единица наследственности живых организмов, представляющая собой участок молекулы ДНК, способной к самокопированию. Для описания единицы передачи культурного наследия Р. Докинз использовал термин «мем», подчёркивая способность мема к самовоспроизведению, подобно гену. Таким образом, под мемом понимается базовая единица информации, значимая в контексте рассматриваемой области культуры (идея, символ, манера или образ действия, осознанно или неосознанно передаваемые от человека к человеку посредством речи, письма, видео, ритуалов, жестов и т. д.). Подобно тому, как в биологической эволюции, согласно теории Ч. Дарвина, имеет место изменчивость и выживание сильнейшего, в культурной эволюции мемы рассматриваются как информация, воспроизводящаяся вариационно и селективно. Согласно Р. Докинзу, мемы копируются путём имитации, обучения и других методов и борются за выживание в человеческой памяти, чтобы впоследствии реплицироваться вновь. Мемы развиваются по причине того, что выживают только некоторые вариации культурных единиц. В математическом и компьютерном моделировании, как правило, предполагается, что мемы участвуют в таких процессах, как имитация (imitation), распространение (propagation), экспрессия (expression), передача (transmission), ассимиляция (assimilation), сохранение (retention) и изменчивость (variation) [9]. Используется следующее соответствие терминологии в генетике и меметике: ген–мем, генотип–мемотип, фенотип–социотип.

С целью решения задач оптимизации П. Москато [16] предложил концепцию имитационного моделирования культурной эволюции (эволюции мемов), рассматриваемой как обобщение биологической эволюции (эволюции генов). При этом он обозначил как минимум два отличия между биологической эволюцией и культурной эволюцией: 1) индивиды не могут выбирать для себя гены, однако мемы могут приобретаться намеренно; 2) индивиды могут модифицировать и улучшать свои мемы, чего они не могут делать со своими генами.

В своей статье [16] П. Москато описал структуру меметического алгоритма (memetic algorithm), которая модифицируется и совершенствуется по сегодняшний день. Меметический алгоритм представляет собой гибридизацию популяционных методов поиска (например, таких как генетический алгоритм, эволюционные стратегии, алгоритм роя частиц, алгоритм дифференциальной эволюции и т. д.), алгоритмов локальной оптимизации и проблемно-ориентированных конструктивных эвристик (см., например, [9, 18, 19]). В русскоязычной литературе меметические алгоритмы также известны под названиями «генетический алгоритм локального поиска» и «гибридный эволюционный алгоритм» (см., например, [1, 4, 5]).

Локальная оптимизация может использоваться на различных стадиях меметических алгоритмов: локальные улучшения на этапе инициализации и / или на этапе постпроцессинга, чередование алгоритмов локального поиска с операторами эволюционного поиска. Также разрабатываются меметические алгоритмы, где одновременно сосуществуют несколько взаимодействующих популяций [9].

В современных меметических алгоритмах координация алгоритмических компонент реализуется с помощью различных адаптивных схем: адаптивных правил, использующих оценки разнообразия популяции, вычисленные на основе значений функции приспособленности или расстояний между особями; адаптивных гиперэвристик; самоадаптирующихся алгоритмов; методов машинного обучения и др. [18, 20].

Отметим, что меметические алгоритмы являются представителями методов из достаточно широкой области, называемой меметическими вычислениями (memetic computing) [18]. В меметических вычислениях для решения различных задач используются сложные динамические структуры, состоящие из взаимодействующих модулей (мемов). Здесь мемы рассматриваются как стратегии и / или операторы, действия которых координируются и адаптируются в процессе вычислений. Развитие и динамика мемов осуществляются подобно распространению идей в обществе. В меметических вычислениях сочетаются методы из различных отраслей компьютерных наук (эволюционные вычисления, многокритериальная оптимизация, машинное обучение, приближенные алгоритмы и т. д.), а также методы из других наук, таких как биология, социология и физика.

Задача коммивояжёра (ЗК) относится к числу классических задач дискретной оптимизации. Дан полный ориентированный граф G с множеством вершин $V = \{v_1, \dots, v_n\}$ и множеством дуг $A = \{(v_i, v_j) : \ v_i, v_j \in V, \ i \neq j\}$. Для каждой дуги $(v_i, v_j) \in A$ задан вес (длина дуги) $c(v_i, v_j) \geqslant 0$. Требуется найти Гамильтонов контур C (маршрут коммивояжёра) с минимальным значением суммарной длины $\sum_{(v_i,v_j)\in C} c(v_i, v_j)$. Если длина дуги (v_i, v_j) не совпадает с длиной обратной дуги (v_j, v_i) по крайней мере для одной пары вершин $v_i, \ v_j$, то задача коммивояжёра называется *несимметричной* (НЗК). Несимметричная задача коммивояжёра возникает в различных практических приложениях: задачи составления расписаний с учётом переналадки оборудования, задачи маршрутизации транспортных средств, задачи цехового типа, распределение товаров и услуг, выпуск продукции и др.

Несимметричная задача коммивояжёра является NP-трудной в сильном смысле [14], поэтому имеет смысл разработка эвристических алгоритмов и метаэвристик, в частности меметических алгоритмов. В настоящей работе предлагается меметический алгоритм для рассматриваемой задачи. Начальная популяция строится с помощью эвристики локального поиска и конструктивной эвристики В. Занга [27]. В процессе эволюции популяции используются проблемно-ориентированная мутация и оптимальная рекомбинация [12]. Баланс между интенсивностью использования локальной оптимизации и интенсивностью использования популяционного поиска регулируется с помощью адаптивного перезапуска.

В операторе кроссинговера комбинируются элементы родительских решений при построении решений потомков. Задача оптимальной рекомбинации (ЗОР) состоит в отыскании наилучшего возможного результата кроссинговера для заданной пары родительских решений задачи оптимизации. ЗОР является вспомогательной задачей и формулируется с учётом основных принципов оператора кроссинговера [22]. С точки зрения меметики оптимальная рекомбинация соответствует такой передаче культурной информации от родителей (наставников) к потомкам, при которой потомок формирует комбинацию мемов (идей, манер или образов действий и мыслей) от родителей согласно своим предпочтениям и интересам, а также ценностям (устоям) общества.

Результаты экспериментальных исследований, представленные в работах [25] и [26], показывают, что операторы кроссинговера, решающие ЗОР приближённо, могут эффективно применяться в генетических алгоритмах для задач на перестановках. ЗОР для несимметричной задачи коммивояжёра является NP-трудной и сводится к задаче коммивояжёра на ориентированном кубическом графе с предписанными дугами [2]. Последняя задача в свою очередь решается с помощью модификации алгоритма Д. Эппштейна [11].

Статья имеет следующую структуру. В разделе 1 представлены общая схема меметического алгоритма и его подробное описание. В разделе 2 описывается предложенный меметический алгоритм для задачи коммивояжёра. Результаты вычислительного эксперимента приведены в разделе 3.

1. Меметический алгоритм

Меметический алгоритм есть алгоритм специальной структуры, где популяция пробных решений развивается с помощью вероятностных операторов селекции, кроссинговера (рекомбинации), мутации, а также методов локального поиска и механизмов обновления популяции. Приведем общую схему современного меметического алгоритма и рассмотрим детально его шаги.

Алгоритм 1. Меметический алгоритм.
Шаг 1. Сгенерировать начальную популяцию P_0, положить $t := 0$.
Шаг 2. Пока не выполнен критерий остановки, выполнять шаги 2.1–2.4
2.1. Построить популяцию потомков Q_t с помощью операторов селекции, кроссинговера и мутации.
2.2. Реализовать локальное улучшение популяции Q_t.
2.3. Сформировать популяцию следующего поколения P_{t+1} из популяций P_t и Q_t, положить $t := t + 1$.
2.4. Если выполняется критерий перезапуска, то осуществить перезапуск алгоритма с новой начальной популяцией P_0, положить $t := 0$.

Начальная популяция строится, как правило, одним из двух способов: особи генерируются случайно, или используются «перспективные» решения, построенные с помощью конструктивных эвристик и / или эвристик локального поиска.

Критерием остановки может быть заданное число итераций или количество вычислений целевой функции, достижение максимального числа итераций без улучшения рекорда, реализация некоторого числа перезапусков алгоритма, достижение заданного значения целевой функции и др.

При построении потомков в первую очередь осуществляется отбор родительских особей. Такой отбор может реализовываться на основе функции приспособленности особей (совпадает с целевой функцией или является её обобщением / модификацией с целью направленного поиска), тогда более приспособленные особи имеют больше шансов быть выбранными в качестве родителей. Примерами являются методы рулетки и турнирного отбора [7]. Также родительские особи могут выбираться на основе разнообразия популяции (среднее расстояние Хэмминга, энтропия Шеннона и др.). В данном случае особи из различных областей поиска используются в роли родителей. Далее родительские особи комбинируются с помощью операторов скрещивания и мутации. Выбор того или иного оператора может быть сделан на этапе создания алгоритма или осуществляется в процессе его выполнения с помощью механизмов обучения, свойств родителей, степени разнообразия популяции и др. Более того, значения настраиваемых параметров меметического алгоритма, таких как вероятности скрещивания и мутации, размер популяции или подпопуляции, могут адаптироваться в процессе работы алгоритма.

Важную роль в меметическом алгоритме играет локальная оптимизация, целью которой является улучшение качества потомков. Здесь потомок подвергается усовершенствованию, что соответствует обучению в жизнедеятельности (улучшение / модификация мемов). Меметический алгоритм объединяет в себе две методологии поиска: популяционные алгоритмы и методы локального поиска, причём может рассматриваться как подход, в котором локальный поиск действует на совокупности решений, с периодическим привлечением операций рекомбинации с целью кооперации и обмена информацией между решениями.

Современные меметические алгоритмы обладают набором методов локального поиска, где определяется, к каким решениям какие операторы применять, с какой интенсивностью и как часто. Для этого используются адаптивные гиперэвристики, механизмы машинного обучения, самоадаптирующиеся и коэволюционные методы, схемы, базирующиеся на разнообразии популяции, и др. Как правило, среди эвристик локального поиска присутствуют случайный локальный поиск и вариации алгоритмов локальной оптимизации с различными ограничениями на максимальное число итераций и долю просматриваемых решений в окрестности. Случайный локальный поиск и локальный спуск в окрестности текущего решения характерны для применения на начальном этапе и в процессе эволюции популяции, тогда как интенсивный локальный поиск (с просмотром большей части или всей окрестности решения на каждой итерации) активно используется на завершающем этапе работы алгоритма.

При отборе особей для следующего поколения принимаются во внимание качество, разнообразие и возраст особей. Важно следить за разнообразием популяции для того, чтобы избежать преждевременного достижения локального оптимума и перейти к освоению новых областей поиска. Различают популяционную схему воспроизведения, где число потомков на одной итерации близко или совпадает с числом особей предыдущего поколения, и стратегию с частичным воспроизведением, где замещается только часть популяции [7]. Также известны меметические алгоритмы, где популяция хранится и обновляется в соответствии с некоторой структурой (см., например, тернарное дерево [8], кольцо [6]). Более того, в эволюционных вычислениях известны (μ, λ) и $(\mu + \lambda)$ эволюционные стратегии, схемы обновления популяции которых могут быть перенесены в меметические алгоритмы.

Если популяция вырождается, что проявляется, например, в снижении степени разнообразия популяции до некоторого порога или длительном неулучшении рекорда, то имеет смысл его перезапуск. При этом строится новая начальная популяция, которая в отдельных случаях содержит часть особей последней популяции, возможно, подверженных изменениям с помощью эвристик.

Также необходимо отметить, что важную роль при разработке меметического алгоритма играет способ представления решений и типы окрестностей, используемых в процедурах локального поиска. Это определяет эффективность, трудоёмкость и перспективность алгоритма.

Если меметический алгоритм можно рассматривать как метод с определённой структурой и специфическими особенностями, то меметические вычисления представляют собой подход к разработке алгоритмов. Меметические вычисления рассматривают алгоритмы как эволюционирующие структуры, состоящие из действующих согласованно и в тоже время конкурирующих операторов (мемов). Это ведёт к идее автоматического построения алгоритмов решения задач путём «грамотной» комбинации и адаптации операторов (мемов).

2. Меметический алгоритм для задачи коммивояжёра

Предлагаемый меметический алгоритм ($\mathcal{MA}$) основан на схеме развития популяции особей в генетическом алгоритме. Каждая особь соответствует решению задачи (фенотипу), которое представлено в алгоритме как строка символов из некоторого конечного алфавита (генотип). Компоненты строк называются генами. $\mathcal{MA}$ использует стратегию управления популяцией, известную как *стационарная схема воспроизведения* [7]. Схема алгоритма имеет следующий вид.

Алгоритм 2. Меметический алгоритм $\mathcal{MA}$ для ЗК.
Шаг 1. Построение начальной популяции с последующим локальным улучшением каждой особи.
Шаг 2. Пока не сработал критерий остановки, выполнять шаги 2.1–2.4.
2.1. Выбрать из популяции двух родителей $\mathbf{p}_1, \mathbf{p}_2$.
2.2. Применить к $\mathbf{p}_1$ и $\mathbf{p}_2$ оператор мутации.
2.3. Построить потомка $\mathbf{p}'$ от $\mathbf{p}_1$, $\mathbf{p}_2$ с помощью оператора кроссинговера.
2.4. Заменить худшую особь в популяции потомком $\mathbf{p}'$.
Шаг 3. Выполнить локальное улучшение каждой особи последней популяции.

Численность популяции N является фиксированной от начала и до конца работы алгоритма $\mathcal{MA}$. Две особи начальной популяции строятся с помощью жадной конструктивной эвристики, предложенной В. Зангом [27]. Оставшиеся $(N-2)$ особей начальной популяции строятся с помощью эвристики «случайного присоединения к контуру» [26]. Особи начальной и последней популяций улучшаются посредством применения эвристики локального поиска, основанного на 3-opt окрестности (см. раздел 2.3).

В качестве оператора селекции используется s-турнирная селекция [7]. Мутация применяется к каждому из родительских решений с вероятностью p_{mut}, которая является настраиваемым параметром алгоритма $\mathcal{MA}$. Реализовано два оператора мутации, которые выполняют случайный спуск в 3-opt или 4-opt окрестности (см. рис. 1). Операторы используются для мутации с равной вероятностью. В операторе скрещивания решается задача оптимальной рекомбинации с помощью алгоритма, представленного в разделе 2.1.

2.1. Оператор рекомбинации

Для заданной пары родительских особей задача оптимальной рекомбинации (ЗОР) состоит в отыскании наилучшего возможного потомка как результата оператора скрещивания при условии, что значение каждого гена потомка копируется из значений соответствующих генов одного или другого родителя (свойство передачи генов) [22].

Решения несимметричной задачи коммивояжёра кодируются списками смежностей, где для каждой вершины указывается непосредственно предшествующая ей вершина. При указанной кодировке решений ЗОР направлена на поиск кратчайшего маршрута коммивояжёра, совпадающего с двумя заданными родительскими решениями по тем дугам, по которым проходят оба родительских контура, и не проходящего по дугам, отсутствующим в обоих из них.

ЗОР при наследовании свойства смежности для несимметричной ЗК является NP-трудной в сильном смысле задачей [2]. Данная ЗОР сводится к ЗК на ориентированных графах с ограниченной степенью вершин и может быть решена за время $O(n2^{\frac{n}{2}})$ [2].

Рассмотрим несимметричную задачу коммивояжёра с предписанными дугами. Дан ориентированный граф с произвольными неотрицательными весами (длинами) дуг и заданным множеством предписанных дуг. Требуется найти кратчайший гамильтонов контур, проходящий по всем предписанным дугам. Степенью вершины v называется общее число дуг, исходящих из вершины v или входящих в вершину v. Кубическим графом называется граф, степень каждой вершины которого не превосходит 3.

Обозначим через A_1 и A_2 множества дуг, по которым проходят родительские маршруты $\mathbf{p}_1$ и $\mathbf{p}_2$. ЗОР для $\mathbf{p}_1$ и $\mathbf{p}_2$ можно рассматривать как несимметричную задачу коммивояжёра на графе $G' = (V, A_1 \cup A_2)$ с множеством предписанных дуг $F' := A_1 \cap A_2$. Заметим, что степень любой вершины графа G' не превосходит четырёх. Несимметричная задача коммивояжёра в таком ориентированном графе эквивалентна несимметричной задаче коммивояжёра на кубическом ориентированном графе $G'' = (V'', A'')$ с множеством предписанных дуг F'', где каждая вершина степени четыре $v \in V$ заменена двумя вершинами $\bar{v}, \tilde{v}$, соединёнными фиктивной дугой $(\bar{v}, \tilde{v})$ нулевого веса. При этом в вершину $\bar{v}$ входят дуги, входившие в v, а из вершины $\tilde{v}$ выходят дуги, выходившие из v. Множество F'' состоит из предписанных дуг F' и фиктивных дуг.

Алгоритм из [2] перебирает все допустимые решения несимметричной задачи коммивояжёра на графе $G'' = (V'', A'')$ с множеством предписанных дуг F'', и для каждого маршрута значение целевой функции вычисляется за линейное время. Число допустимых решений есть $O(2^{(|A''|-|F''|)/4})$. Таким образом данный подход позволяет решить ЗОР за время $O(2^{d/4}n)$, где $d \leqslant 2n$ есть число дуг, содержащихся только в одном из родительских контуров. В настоящей работе используется модификация алгоритма из [2], где целевая функция вычисляется за время $O(1)$, что гарантирует общую трудоёмкость $O(2^{\frac{n}{2}})$.

В большинстве из известных генетических алгоритмов для задачи коммивояжёра итерационно используется локальная оптимизация после операторов кроссинговера (см., например, [8, 15, 25]). Оптимальная рекомбинация в свою очередь может рассматриваться как локальный спуск в окрестности, определяемой парой родительских решений. Поэтому в алгоритме $\mathcal{MA}$ локальный поиск применяется только на начальной и финальной стадиях.

2.2. Эвристика локального поиска

Окрестность k-opt в задаче коммивояжёра состоит из всех гамильтоновых контуров, получающихся из данного маршрута заменой k дуг. Используемая в алгоритме $\mathcal{MA}$ эвристика локального поиска основана на просмотре подмножества контуров в окрестности 3-opt.

Предпринимается попытка улучшить заданный маршрут путём замены трёх дуг. В примере, представленном слева на рис. 1, дуги (v_{i_1}, v_{i_2}), (v_{i_4}, v_{i_3}) и (v_{i_6}, v_{i_5}) удаляются, а дуги (v_{i_1}, v_{i_3}), (v_{i_4}, v_{i_5}) и (v_{i_6}, v_{i_2}) добавляются. В качестве кандидатов для (v_{i_1}, v_{i_2}) рассматриваются все дуги текущего маршрута в порядке убывания их весов. Варианты для вершины v_{i_3} достаточно искать только среди тех вершин, которые ближе к вершине v_{i_1} чем v_{i_2}. Поэтому для каждой вершины v предварительно формируется список из остальных вершин в порядке возрастания весов дуг от вершины v. Кандидаты на вершину v_{i_3} просматриваются согласно списку для v_{i_1}, пока не будет достигнута вершина u, для которой $c_{v_{i_1},u} \geqslant c_{v_{i_1},v_{i_2}}$. Более того, только $\lceil 0.2n \rceil$ ближайших вершин хранится в отсортированном списке каждой вершины, что позволяет уменьшить время работы и объём используемой памяти в алгоритме (см., например, [15]). Отметим, что при добавлении дуги (v_{i_1}, v_{i_3}) образуется замкнутый контур C. Среди всех вершин этого контура в качестве v_{i_5} выбирается вершина, приводящая после замены трёх дуг к наиболее эффективному маршруту (вершины v_{i_4} и v_{i_6} идентифицируются однозначно). Эвристика локального поиска останавливается, если никакая замена трёх дуг не приводит к улучшению текущего маршрута. В противном случае описанная процедура повторяется для нового контура.

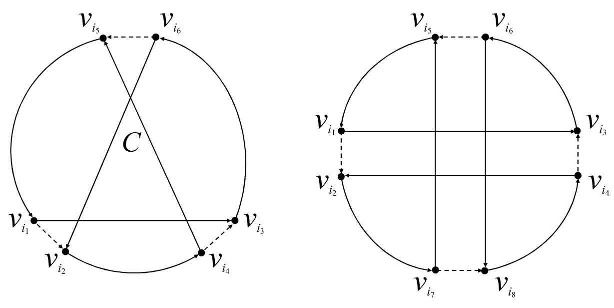

Рис. 1. Окрестности 3-opt и 4-opt

Для уменьшения времени работы представленной эвристики локального поиска используются такие известные стратегии, как «don't look bits» (не просматривай вершины, приводившие к ухудшению) и «first improving move» (переходи в первый улучшающий маршрут) из работы [15].

2.3. Механизмы перезапуска

Баланс между локальным и эволюционным методами поиска настраивается адаптивно. Одним из простых способов адаптации является следующее классическое правило перезапуска: алгоритм $\mathcal{MA}$ выполняет $N/2$ итераций без перезапуска. Далее $\mathcal{MA}$ перезапускается, как только будет достигнута итерация, номер которой в два раза больше номера итерации, где получено последнее улучшение рекордного значения целевой функции.

В работе также используется правило перезапуска, основанное на методе переписи Шнабеля. Метод переписи Шнабеля был разработан в биометрии для статистической оценки размера популяции животных. Согласно данному методу, из популяции извлекаются выборки размера n_0 и подсчитывается количество *различных* животных, наблюдаемых в выборках. При этом предполагается, что вероятность отлова для всех животных одинакова. Отобранные животные помечаются, если это не было сделано ранее, и возвращаются обратно в популяцию. Далее статистическая оценка для общего числа ν особей в популяции вычисляется на основе общего числа отловленных и помеченных животных. Мы применяем метод переписи Шнабеля для оценки количества значений, которые дискретная случайная величина может принимать с ненулевой вероятностью, аналогично работе [3].

Пусть величина r задаёт длину исторического периода, рассматриваемого для статистического анализа в правиле перезапуска алгоритма $\mathcal{MA}$. Предполагается, что в течение последних r итераций все новые потомки в $\mathcal{MA}$ имеют одинаковое распределение вероятностей и могут рассматриваться как отобранные животные в методе переписи Шнабеля. В свою очередь указанный метод используется для оценки числа ν различных решений, которые можно посетить с положительной вероятностью в предположении, что текущее распределение потомков остаётся неизменным.

Также предполагается, что выборка, наблюдаемая за последние r итераций алгоритма $\mathcal{MA}$, состоит из r независимых решений-потомков. Определим случайную величину K как число различных решений в этой выборке. Делается ещё одно упрощающее предположение о том, что все решения, которые могут быть сгенерированы с текущим распределением, имеют равные вероятности. Тогда для любого фиксированного ν случайная величина K имеет следующее распределение:

$$\Pr\{K = k\} = \frac{\nu!}{(\nu - k)!}\frac{S(r,k)}{\nu^r},$$

где $S(r,k) = \frac{1}{k!}\sum_{s=0}^{k}(-1)^k\binom{k}{s}(k-s)^r$ — это число Стирлинга второго рода. Данное распределение также известно как распределение Арфведсона. Тогда оценка максимального правдоподобия $\hat{\nu}^{\mathrm{ML}}$ для неизвестной величины ν равна

$$\hat{\nu}(r,k) = \mathrm{argmax}\left\{\frac{\nu!}{(\nu-k)!\nu^r}\right\}, \tag{1}$$

где k — это общее количество решений сгенерированных на последних r итерациях. Значение величины $\hat{\nu}^{\text{ML}} = \hat{\nu}(r,k)$ может быть вычислено из (1) с помощью стандартных методов одномерной оптимизации. Технические подробности и дополнительные ссылки на литературу можно найти в [23].

В настоящей статье используется правило перезапуска из [13], согласно которому алгоритм перезапускается, как только оценка $\hat{\nu}^{\text{ML}}$ становится равной k. Обоснование такого правила заключается в том, что выполнение равенства $\hat{\nu}^{\text{ML}} = k$ свидетельствует, скорее всего, об отсутствии непосещённых решений в той части допустимой области, где популяция алгоритма находилась последние r итерации. В такой ситуации целесообразно перезапустить алгоритм, а не ждать пока распределение популяции существенно изменится под влиянием эволюционных процессов.

Значение параметра r настраивается в процессе работы алгоритма. Всякий раз, когда значение рекорда уменьшается, величина r полагается равной размеру популяции N. Если же за последние $2r$ итераций значение рекорда не изменяется, то r увеличивается в два раза. Отметим, что улучшение рекорда говорит о достижении популяцией новой неисследованной области поиска, поэтому длину исторического периода следует уменьшить (в предлагаемом варианте до размера популяции N). С целью уменьшения времени работы алгоритма критерий остановки проверяется только при обновлении значения r. При многопроцессорной реализации алгоритма эта проверка может выполняться на каждой итерации $\mathcal{MA}$.

3. Вычислительный эксперимент

Предлагаемый алгоритм $\mathcal{MA}$ реализован на языке Java и тестировался на ЭВМ Intel Xeon E5420 2.5 ГГц, оперативная память 16 ГБ.

Алгоритм $\mathcal{MA}$ сравнивается с одним из наиболее конкурентоспособных генетических алгоритмов решения НЗК, предложенным в работе [17]. Алгоритм из [17] (GA_{EAX}) использует оператор скрещивания (EAX), основанный на схеме A-B-циклов, где при построении потомка сначала формируются ориентированные подциклы с чередующимися дугами от одного и другого родителей, а затем из подциклов строится гамильтонов контур. Данный оператор допускает наличие у потомка дуг, отсутствующих в родительских маршрутах. GA_{EAX} основан на популяционной схеме воспроизведения (размер популяции равен 300) и при построении начальной популяции применяет локальную оптимизацию на основе 3-opt окрестности. Этот алгоритм в [17] реализован на языке C++ и тестировался на ЭВМ Intel Xeon 2.93 ГГц.

Алгоритмы $\mathcal{MA}$ и GA_{EAX} запускались на каждой тестовой задаче 100 раз на сопоставимое время, а именно алгоритму $\mathcal{MA}$ отводилось в 1,465 раза большее времени, чем зафиксированное время счета GA_{EAX}. Указанный множитель выбран, исходя из соотношения тактовых частот процессоров и того факта, что Java-компилятор оценивается на 25 % медленнее C++-компилятора.

Вычислительный эксперимент проведён на 126 тестовых примерах с числом вершин от 64 до 315 из библиотеки [24]. Эти примеры характеризуются тем свойством, что длины дуг несимметричны, но близки по значению друг к другу. Алгоритм $\mathcal{MA}$ тестировался с двумя рассматриваемыми в настоящей работе правилами перезапуска и при двух размерах популяции ($N = 100$ и $N = 300$). Лучшие результаты показали версии алгоритма $\mathcal{MA}$ с $N = 300$, причём классическое правило перезапуска несколько уступает правилу перезапуска, основанному на методе переписи Шнабеля. Обозначим через C_{best} и C_{aver} лучшее и среднее значения целевой функции за 100 запусков. Пусть Δ_{aver} соответствует относительному отклонению в % величины C_{aver} от оптимального значения критерия оптимизации. Результаты вычислительного эксперимента для алгоритмов $\mathcal{MA}_{SC300}$ ($N = 300$, и перезапуск основан на методе переписи Шнабеля) и GA_{EAX} представлены в таблицах 1–3 в приложении. Для проверки наличия статистической значимости различий между результатами анализируемых методов применялся тест суммы рангов Уилкоксона с уровнем значимости $\alpha = 0,05$. Для сравнения использовались значения величин C_{best} и C_{aver} по всем тестовым примерам.

Среднее отклонение от оптимума Δ_{aver} для $\mathcal{MA}_{SC300}$ составляет 0,0005, что почти в 1,5 раза больше, чем для алгоритма GA_{EAX}. Однако со 100 % частотой получения оптимума алгоритмом $\mathcal{MA}_{SC300}$ было решено 82 задачи, а алгоритмом GA_{EAX} 68 задач (см. значения C_{aver}, выделенные полужирным шрифтом в таблицах приложения). Значимых отличий между средними значениями целевой функции C_{aver} для алгоритмов $\mathcal{MA}_{SC300}$ и GA_{EAX} не выявлено. Идентичное утверждение справедливо для лучших найденных значений критерия C_{best}.

В работе [12] проводилось сравнение по частоте получения оптимума версии алгоритма $\mathcal{MA}$ (использовалось классическое правило перезапуска и отсутствовала локальная оптимизация на этапе постпроцессинга) с алгоритмом GA_{EAX} при $N = 100$ на несимметричных задачах библиотеки TSPLIB. Были сделаны аналогичные выводы о схожести в результатах работы алгоритмов. Отметим, что большинство примеров библиотеки TSPLIB (кроме серии rbg) характеризуются более существенными отличиями в длинах дуг, чем в рассматриваемых здесь примерах. Причём на серии rbg оптимумы легко вычисляются на начальном этапе работы $\mathcal{MA}$ с помощью алгоритма В. Занга [27], что не характерно для задач из библиотеки [24]. Таким образом, можно сделать вывод о конкурентоспособности алгоритмов $\mathcal{MA}$ и GA_{EAX} на задачах различной структуры.

4. Заключение

Представлен популяционный алгоритм для несимметричной задачи коммивояжёра, основанный на использовании принципов меметики. Результаты эксперимента на известных тестовых примерах свидетельствуют об эффективности использования оптимальной рекомбинации в операторах кроссинговера и о перспективности разработанного меметического алгоритма. Также продемонстрирована важность соблюдения баланса между локальным и эволюционным методами поиска. Дальнейшие исследования могут быть направлены на внедрение в предложенный меметический алгоритм различных по свойствам операторов кроссинговера и локальной оптимизации, а также их адаптивной настройки в процессе эволюции.

Благодарности

Данная работа была выполнена при поддержке РНФ, проект № 17-18-01536.

Приложение

Настоящее приложение содержит таблицы с результатами вычислительного эксперимента для алгоритмов $\mathcal{MA}_{SC300}$ и GA_{EAX} на тестовых примерах из библиотеки [24]. Здесь C_{best} и C_{aver} есть лучшее и среднее значения целевой функции за 100 запусков, Δ_{aver} — относительное отклонение в % величины C_{aver} от оптимального значения критерия оптимизации.

Таблица 1. Сравнение результатов работы алгоритмов $\mathcal{MA}_{SC300}$ и GA_{EAX} (часть 1)

задача	$\mathcal{MA}_{SC300}$			GA_{EAX}		
	C_{best}	C_{aver}	Δ_{aver}	C_{best}	C_{aver}	Δ_{aver}
D41140	417880	**417880**	0	417880	**417880**	0
D61440	512674	**512674**	0	512674	**512674**	0
D4940m	336325	**336325**	0	336325	**336325**	0
D81740m	664613	**664613**	0	664613	**664613**	0
D4940	422508	**422508**	0	422508	**422508**	0
D81740	809706	**809706**	0	809706	**809706**	0
D82040m	689971	**689971**	0	689971	**689971**	0
D102240m	803595	**803595**	0	803595	**803595**	0
D82040	790971	**790971**	0	790971	**790971**	0
D102240	1027751	**1027751**	0	1027751	**1027751**	0
D102640m	815096	**815096**	0	815096	**815096**	0
D122640m	880335	**880335**	0	880335	**880335**	0
D122940m	997772	**997772**	0	997772	**997772**	0
D102640	926142	**926142**	0	926142	**926142**	0
D122640	990448	**990448**	0	990448	**990448**	0
D122940	1316922	1316923,48	0,00011	1316922	1316922,2	0,00002
D143040a	1072673	**1072673**	0	1072673	**1072673**	0
D143340	1452339	**1452339**	0	1452339	**1452339**	0
D143340m	1100177	**1100177**	0	1100177	**1100177**	0
D143040	1472328	1472344,1	0,0011	1472328	1472332,6	0,00031
D143040b	1210175	**1210175**	0	1210175	**1210175**	0
D163440a	1145914	**1145914**	0	1145914	**1145914**	0

Таблица 2. Сравнение результатов работы алгоритмов $\mathcal{MA}_{SC300}$ и GA_{EAX} (часть 2)

задача	$\mathcal{MA}_{SC300}$			GA_{EAX}		
	C_{best}	C_{aver}	Δ_{aver}	C_{best}	C_{aver}	Δ_{aver}
D163440	1452914	**1452914**	0	1452914	**1452914**	0
D163440b	1310767	1310767,2	0,000015	1310767	1310767,3	0,00002
D183840a	1255781	**1255781**	0	1255781	**1255781**	0
D163740	1468709	**1468709**	0	1468709	**1468709**	0
D183840	1652091	**1652091**	0	1652091	**1652091**	0
D183840b	1449912	**1449912**	0	1449912	**1449912**	0
D204240a	1438448	**1438448**	0	1438448	1438453,5	0,00038
D184040a	1331558	**1331558**	0	1331558	**1331558**	0
D204240	1980807	1980813,79	0,00034	1980807	1980814,5	0,00038
D184040	1686092	**1686092**	0	1686092	**1686092**	0
D184040b	1493759	**1493759**	0	1493759	1493776,2	0,00115
D204240b	1764636	**1764636**	0	1764636	**1764636**	0
D41141	454381	**454381**	0	454381	**454381**	0
D4941	452101	**452101**	0	452101	452107,4	0,001416
D61441	561657	**561657**	0	561657	**561657**	0
D61641	633760	**633760**	0	633760	**633760**	0
D81741	730893	730893,14	0,000019	730893	730893,9	0,00012
D82041m	804695	**804695**	0	804695	**804695**	0
D102241m	854290	**854290**	0	854290	**854290**	0
D82041	80751	**880751**	0	880751	**880751**	0
D102241	1029382	**1029382**	0	1029382	**1029382**	0
D122641a	1011650	**1011650**	0	1011650	**1011650**	0
D102641m	1009032	**1009032**	0	1009032	**1009032**	0
D122941m	1058436	**1058436**	0	1058436	**1058436**	0
D122641	1306865	**1306865**	0	1306865	**1306865**	0
D102641	1199118	**1199118**	0	1199118	1199121,2	0,00026
D122941	1243493	1243500,01	0,00056	1243493	**1243493**	0
D122641b	1156769	**1156769**	0	1156769	**1156769**	0
D143041a	1103917	1103917,9	0,000082	1103917	1103917,2	0,000018
D143341a	1201011	**1201011**	0	1201011	1201016	0,000416
D143341b	1376037	1376048,01	0,0027	1376011	1376020,6	0,000698
D143041	1389319	1389320,07	0,0019	1389296	1389296	0,000144
D143341	1584011	1584020,66	0,00061	1584011	1584022,4	0,000720
D143041b	1219063	**1219063**	0	1219063	**1219063**	0
D163441a	1313082	**1313082**	0	1313082	**1313082**	0
D163441	1761528	1761533,13	0,00029	1761528	1761554,6	0,0015
D163441b	1547424	**1547424**	0	1547424	**1547424**	0
D163741m	1343095	1343150,94	0,0042	1343095	**1343095**	0

Таблица 3. Сравнение результатов работы алгоритмов $\mathcal{MA}_{SC300}$ и GA_{EAX} (часть 3)

задача	$\mathcal{MA}_{SC300}$			GA_{EAX}		
	C_{best}	C_{aver}	Δ_{aver}	C_{best}	C_{aver}	Δ_{aver}
D163741	1530323	1530325,18	0,00014	1530323	1530329,7	0,000438
D183841a	1385166	**1385166**	0	1385166	**1385166**	0
D183841	1769648	1769648	0,00033	1769648	1769648	0,000339
D183841b	1552385	1552385	0,00038	1552385	1552385	0,000387
D184041b	1591521	1591531,1	0,00063	1591521	**1591521**	0
D184041a	1491653	**1491653**	0	1491653	**1491653**	0
D184041	1803657	**1803657**	0	1803657	**1803657**	0
D204241a	1510820	1510824,8	0,00032	1510820	1510820,9	0,00006
D204441a	1556414	**1556414**	0	1556414	**1556414**	0
D204241	1872381	1872412,26	0,00268	1872362	1872363,1	0,000059
D204441	2007883	2008038,82	0,01468	2007744	2007778,5	0,001718
D204241b	1693166	1693170,2	0,00025	1693166	1693169,9	0,000230
D204441b	1785585	1785605,29	0,00114	1785585	1785588,3	0,000185
D4942	517768	**517768**	0	517768	**517768**	0
D61642	875072	**875072**	0	875072	**875072**	0
D61442	768117	**768117**	0	768117	**768117**	0
D81742	828368	**828368**	0	828368	**828368**	0
D82042	849394	**849394**	0	849394	**849394**	0
D102242	963636	963636,98	0,0001	963636	963636,6	0,000062
D122642m	1077563	1077566,15	0,00085	1077557	1077562,4	0,000501
D122942m	1149684	1149696,49	0,0023	1149670	1149674,2	0,000365
D102642	1065324	**1065324**	0	1065324	1065326,2	0,000207
D122642	1153625	**1153625**	0	1153625	1153629,6	0,000399
D122942	1235878	**1235878**	0	1235878	1235882,4	0,000356
D143042m	1224644	**1224644**	0	1224644	1224664,1	0,001641
D143342m	1329993	1329994,5	0,0001	1329993	1329995,5	0,000188
D143042	1369730	1369882,7	0,0111	1369759	1369763,5	0,002446
D143342	1525120	1525123,2	0,0002	1525122	1525122	0,000131
D163442	1623001	1623005,2	0,0019	1622974	1622984,5	0,000647
D163442a	1367607	**1367607**	0	1367607	**1367607**	0
D163742a	1432892	1432917	0,0017	1432892	1432921,6	0,002066
D163442b	1442855	1442890,6	0,0024	1442855	1442857	0,000139
D163742b	1602916	**1602916**	0	1602916	1602952,5	0,002277
D163742	1888149	1888160,8	0,0006	1888149	1888173,4	0,001292
D183842m	1434387	1434407,2	0,0014	1434387	1434407,1	0,001401
D183842	1542727	1542747,5	0,0013	1542727	1542740,9	0,000901
D184042m	1534827	**1534827**	0	1534827	**1534827**	0
D184042	1753907	**1753907**	0	1753907	**1753907**	0

Таблица 4. Сравнение результатов работы алгоритмов $\mathcal{MA}_{SC300}$ и GA_{EAX} (часть 4)

задача	$\mathcal{MA}_{SC300}$			GA_{EAX}		
	C_{best}	C_{aver}	Δ_{aver}	C_{best}	C_{aver}	Δ_{aver}
D204242a	1587087	1587090,9	0,0002	1587087	1587089,3	0,000145
D204242	1872624	**1872624**	0	1872624	1872626,1	0,000112
D204242b	1742305	1742310,8	0,0003	1742305	1742312	0,000402
D41143	695196	**695196**	0	695196	**695196**	0
D61443	822217	822217,8	0,0001	822217	822217,9	0,000109
D61643	869353	**869353**	0	869353	**869353**	0
D81743	948708	948718,1	0,0011	948708	948719,1	0,001170
D82043m	1003444	**1003444**	0	1003444	1003444,2	0,00002
D82043	1079720	**1079720**	0	1079720	**1079720**	0
D102643	1121130	**1121130**	0	1121130	**1121130**	0
D122643	1218825	**1218825**	0	1218825	**1218825**	0
D122943m	1365993	**1365993**	0	1365993	**1365993**	0
D122943	1526097	**1526097**	0	1526097	**1526097**	0
D4944	758785	758786,9	0,0003	758785	758791	0,000791
D61444	964730	964737,2	0,0007	964730	964740,4	0,001078
D61644	984935	984936	0,0001	984935	984937	0,000203
D81744	995860	**995860**	0	995860	995865,8	0,000582
D102244m	1245433	1245436,1	0,0002	1245433	1245436,6	0,000289
D82044	1091866	**1091866**	0	1091866	**1091866**	0
D122644m	1295659	**1295659**	0	1295659	1295663,4	0,000340
D122944m	1380447	**1380447**	0	1380447	1380447,9	0,000065
D122944	1456564	**1456564**	0	1456564	1456573,8	0,000673
D102644	1379555	**1379555**	0	1379555	1379562,2	0,000522
D143044m	1438672	1438672,4	0,00002	1438672	1438704,1	0,002231
D143044	1578725	1578745	0,0012	1578725	1578759,7	0,002198
D143344	1549308	**1549308**	0	1549308	**1549308**	0
D163744m	1671485	**1671485**	0	1671485	**1671485**	0
D163744	1801619	1801622,3	0,0002	1801619	1801621,8	0,000155
среднее	1229537,1	1229542,6	0,0005	1229535,35	1229539,1	0,0003

Литература

1. Алексеева Е.В., Кочетов Ю.А. Генетический локальный поиск для задачи о p-медиане с предпочтениями клиентов // Дискретн. анализ и исслед. опер. Сер. 2. 2007. Т. 14, № 1. С. 3–31.
2. Еремеев А.В. О сложности оптимальной рекомбинации для задачи коммивояжёра // Дискретн. анализ и исслед. опер. 2011. Т. 18, № 1. С. 27—40.
3. Еремеев А.В., Ривс К.Р. О доверительных интервалах для числа локальных оптимумов // Математические структуры и моделирование. 2017. Вып. 41. С. 55–74.
4. Кочетов Ю.А., Плясунов А.В. Генетический локальный поиск для задачи о разбиении графа на доли ограниченной мощности // Ж. вычисл. матем. и матем. физ. 2012. Т. 52, № 1. С. 164-–176.
5. Письменная В.А. Алгоритмическое и программное обеспечение меметического алгоритма поиска условного глобального экстремума // Электронный журнал «Труды МАИ». 2015. Вып. 79. С. 1–27.
6. Плотников О.А., Подвальный Е.С. Разработка меметического алгоритма для решения задач оптимального планирования грузоперевозок // Вестник Воронежского государственного технического университета. 2012. Т. 8, № 10-1. С. 19–24.
7. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечёткие системы. М. : Горячая линия–Телеком, 2006. 452 с.
8. Buriol L.S., Franca P.M., Moscato P. A new memetic algorithm for the asymmetric traveling salesman problem // Journal of Heuristics. 2004. V. 10. P. 483–506.
9. Chen X., Ong Y-S., Lim M-H., Chen Tan K. A multi-facet survey on memetic computation // IEEE Transactions on Evolutionary Computation. 2011. V. 15, No. 5. P. 591–607.
10. Dawkins R. The Selfish Gene. Oxford University Press, 1976. 224 p.
11. Eppstein D. The traveling salesman problem for cubic graphs // Journal of Graph Algorithms and Applications. 2007. V. 11, No. 1.
12. Eremeev A.V., Kovalenko Y.V. Genetic algorithm with optimal recombination for the asymmetric travelling salesman problem // Large-Scale Scientific Computing, LNCS, Springer International Publishing, Cham. 2018. V. 10665, P. 341–349.
13. Eremeev A.V. A restarting rule based on the Schnabel census for genetic algorithms // Learning and Intelligent Optimization, LNCS, Springer International Publishing, Cham. 2019. V. 11353. P. 337–351.
14. Garey M.R., Johnson D.S. Computers and Intractability. A Guide to the Theory of NP-completeness. W.H. Freeman and Company, San Francisco, 1979. 340 p.
15. Johnson D.S., McGeorch L.A. The traveling salesman problem: a case study // Local Search in Combinatorial Optimization, John Wiley & Sons Ltd. 1997. P. 215–336.
16. Moscato P. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Technical Report C3P 826, California Institute of Technology, Pasadena, CA, 1989. 68 p.
17. Nagata Y., Soler D. A new genetic algorithm for the asymmetric TSP // Expert Syst. with Applications. 2012. V. 10. P. 8947—8953.
18. Neri F., Cotta C. Memetic algorithms and memetic computing optimization: A literature review // Swarm and Evolutionary Computation. 2012. V. 2. P. 1–14.
19. Neri F., Cotta C., Moscat P. Handbook of Memetic Algorithms. Springer-Verlag, Berlin Heidelberg, 2012. 309 p.

20. Neri F., Toivanen J., Cascella G.L., Ong Y.S. An adaptive multimeme algorithm for designing HIV multidrug therapies // IEEE/ACM Transactions on Computational Biology and Bioinformatics. 2007. V. 4. P. 264–278.
21. Norman M., Moscato P. A competitive and cooperative approach to complex combinatorial search // 20th Joint Conference on Informatics and Operations Research, Buenos Aires, Argentina. 1991. P. 3.15–3.29.
22. Radcliffe N.J. The algebra of genetic algorithms // Annals of Mathematics and Artificial Intelligence. 1994. V. 10, No. 4. P. 339–384.
23. Reeves C.R., Eremeev A.V. Statistical analysis of local search landscapes // J. Oper. Res. Soc. 2004. V. 55, No. 7. P. 687–693.
24. Soler D., Martínez E., Micó J.C. A transformation for the mixed general routing problem with turn penalties // Journal of the Operational Research Society. 2008. V. 59. P. 540—547.
25. Tinós R., Whitley D., Ochoa G. Generalized asymmetric partition crossover (GAPX) for the asymmetric TSP // The 2014 Annual Conference on Genetic and Evolutionary Computation, ACM New York, NY. 2014. P. 501–508.
26. Yagiura M., Ibaraki T. The use of dynamic programming in genetic algorithms for permutation problems // Eur. Jour. Oper. Res. 1996. V. 92. P. 387–401.
27. Zhang W. Depth-first branch-and-bound versus local search: A case study // 17th National Conf. on Artificial Intelligence, Austin, TX. 2000. P. 930–935.

APPLYING PRINCIPLES OF MEMETICS TO SOLVING THE TRAVELING SALESMAN PROBLEM

A.V. Eremeev[1]
Dr.Sc. (Phys.-Math.), e-mail: eremeev@ofim.oscsbras.ru
Yu.V. Kovalenko[1,2]
Ph.D. (Phys.-Math.), e-mail: julia.kovalenko.ya@yandex.ru

[1]Institute of Scientific Information for Social Sciences RAS, Moscow, Russia
[2]Dostoevsky Omsk State University, Omsk, Russia

Abstract. We consider a population-based algorithm for the asymmetric travelling salesman problem, based on the principles of the memetics. The initial population of tentative solutions is built by means of greedy constructive heuristics. The 3-opt local search is used to improve the initial and the final populations. A balance between the intensity of population-based search and the intensity of the local search methods is chosen by the means of adaptive restart rules. The results of computational experiment on the benchmark instances indicate the viability and effectiveness of the proposed algorithm.

Keywords: memes, genetic algorithm, local optimization, computational experiment.

References

1. Alekseeva E.V. and Kochetov Yu.A. Geneticheskii lokal'nyi poisk dlya zadachi o p-mediane s predpochteniyami klientov. Diskretn. analiz i issled. oper., Cer. 2, 2007, vol. 14, no. 1. pp. 3—31. (in Russian)

2. Eremeev A.V. O slozhnosti optimal'noi rekombinatsii dlya zadachi kommivoyazhera. Diskretn. analiz i issled. oper., 2011, vol. 18, no. 1, pp. 27-–40. (in Russian)
3. Eremeev A.V. and Rivs K.R. O doveritel'nykh intervalakh dlya chisla lokal'nykh optimumov. Matematicheskie struktury i modelirovanie, 2017, issue 41, pp. 55–74. (in Russian)
4. Kochetov Yu.A. and Plyasunov A.V. Geneticheskii lokal'nyi poisk dlya zadachi o razbienii grafa na doli ogranichennoi moshchnosti. Zh. vychisl. matem. i matem. fiz., 2012, vol. 52, no. 1, pp. 164–176. (in Russian)
5. Pis'mennaya V.A. Algoritmicheskoe i programmnoe obespechenie memeticheskogo algoritma poiska uslovnogo global'nogo ekstremuma. Elektronnyi zhurnal "Trudy MAI", 2015, issue 79, pp. 1–27. (in Russian)
6. Plotnikov O.A. and Podval'nyi E.S. Razrabotka memeticheskogo algoritma dlya resheniya zadach optimal'nogo planirovaniya gruzoperevozok. Vestnik Voronezhskogo gosudarstvennogo tekhnicheskogo universiteta, 2012, vol. 8, no. 10-1, pp. 19–24. (in Russian)
7. Rutkovskaya D., Pilin'skii M., and Rutkovskii L. Neironnye seti, geneticheskie algoritmy i nechetkie sistemy. Moscow, Goryachaya liniya–Telekom Publ., 2006, 452 p. (in Russian)
8. Buriol L.S., Franca P.M., and Moscato P. A new memetic algorithm for the asymmetric traveling salesman problem. Journal of Heuristics, 2004, vol. 10, pp. 483–506.
9. Chen X., Ong Y-S., Lim M-H., and Chen Tan K. A multi-facet survey on memetic computation. IEEE Transactions on Evolutionary Computation, 2011, vol. 15, no. 5, pp. 591–607.
10. Dawkins R. The Selfish Gene. Oxford University Press, 1976, 224 p.
11. Eppstein D. The traveling salesman problem for cubic graphs. Journal of Graph Algorithms and Applications, 2007, vol. 11, no. 1.
12. Eremeev A.V. and Kovalenko Y.V. Genetic algorithm with optimal recombination for the asymmetric travelling salesman problem. Large-Scale Scientific Computing, LNCS, Springer International Publishing, Cham, 2018, vol. 10665, pp. 341–349.
13. Eremeev A.V. A restarting rule based on the Schnabel census for genetic algorithms. Learning and Intelligent Optimization, LNCS, Springer International Publishing, Cham, 2019, vol. 11353, pp. 337-351.
14. Garey M.R. and Johnson D.S. Computers and Intractability. A Guide to the Theory of NP-completeness. W.H. Freeman and Company, San Francisco, 1979, 340 p.
15. Johnson D.S. and McGeorch L.A. The traveling salesman problem: a case study // Local Search in Combinatorial Optimization, John Wiley & Sons Ltd. 1997. P. 215–336.
16. Moscato P. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Technical Report C3P 826, California Institute of Technology, Pasadena, CA, 1989, 68 p.
17. Nagata Y. and Soler D. A new genetic algorithm for the asymmetric TSP. Expert Syst. with Applications, 2012, vol. 10, pp. 8947–8953.
18. Neri F. and Cotta C. Memetic algorithms and memetic computing optimization: A literature review. Swarm and Evolutionary Computation, 2012, vol. 2, pp. 1–14.
19. Neri F., Cotta C., and Moscat P. Handbook of Memetic Algorithms. Springer-Verlag, Berlin Heidelberg, 2012, 309 p.
20. Neri F., Toivanen J., Cascella G.L., and Ong Y.S. An adaptive multimeme algorithm

for designing HIV multidrug therapies. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2007, vol. 4, pp. 264–278.

21. Norman M. and Moscato P. A competitive and cooperative approach to complex combinatorial search. 20th Joint Conference on Informatics and Operations Research, Buenos Aires, Argentina, 1991, pp. 3.15–3.29.
22. Radcliffe N.J. The algebra of genetic algorithms. Annals of Mathematics and Artificial Intelligence, 1994, vol. 10, no. 4, pp. 339–384.
23. Reeves C.R. and Eremeev A.V. Statistical analysis of local search landscapes. J. Oper. Res. Soc., 2004, vol. 55, no. 7, pp. 687–693.
24. Soler D., Martínez E., and Micó J.C. A transformation for the mixed general routing problem with turn penalties. Journal of the Operational Research Society, 2008, vol. 59, pp. 540–547.
25. Tinós R., Whitley D., and Ochoa G. Generalized asymmetric partition crossover (GAPX) for the asymmetric TSP. The 2014 Annual Conference on Genetic and Evolutionary Computation, ACM New York, NY., 2014, pp. 501–508.
26. Yagiura M. and Ibaraki T. The use of dynamic programming in genetic algorithms for permutation problems. Eur. Jour. Oper. Res., 1996, vol. 92, pp. 387–401.
27. Zhang W. Depth-first branch-and-bound versus local search: A case study. 17th National Conf. on Artificial Intelligence, Austin, TX., 2000, pp. 930–935.

Дата поступления в редакцию: 17.09.2019