Труды МАИ. 2024. № 135
Trudy MAI, 2024, no. 135

Научная статья
УДК 519.6
URL: https://trudymai.ru/published.php?ID=179694

МОДИФИКАЦИЯ АСИНХРОННОГО МЕТОДА МУРАВЬИНЫХ КОЛОНИЙ ДЛЯ ЗАДАЧ ПОИСКА РАЦИОНАЛЬНЫХ РЕШЕНИЙ ПАРАМЕТРИЧЕСКОЙ ЗАДАЧИ

Юрий Павлович Титов[1]✉, Владимир Анатольевич Судаков[2]

[1]Московский авиационный институт (национальный исследовательский университет), Москва, Россия

[2]Институт прикладной математики имени М.В. Келдыша РАН, Москва, Россия

[1]kalengul@mail.ru✉

[2]sudakov@ws-dss.com

Аннотация. Рассматриваются модификации бионической оптимизации методом муравьиных колоний для решения параметрических задач, в частности определения рационального расположения блоков на космическом корабле. Предложены модификации метода, позволяющие производить направленный перебор параметров с целью поиска всех рациональных решений без применения мультистарта. Полный направленный перебор гарантирует нахождение всех оптимальных и рациональных решений, а применение переупорядочивания рассматриваемых решений с помощью модификации метода муравьиных колоний позволяет рассмотреть лучшие решения на ранних этапах. Для практической реализации на кластерной системе в работе рассматриваются асинхронные методы муравьиных колоний, позволяющие

обеспечить параллельное рассмотрение решений на многопоточном кластере. Предложена структура программного обеспечения, позволяющая избежать блокировок кластера, тем самым увеличить эффективность использования процессорного времени.

Ключевые слова: метод муравьиных колоний, метаэвристическая оптимизация, направленный перебор, параметрическая задача, вычислительный кластер, алгоритмы, не использующие блокировки

Для цитирования: Титов Ю.П., Судаков В.А. Модификация асинхронного метода муравьиных колоний для задач поиска рациональных решений параметрической задачи // Труды МАИ. 2024. № 135. URL: https://trudymai.ru/published.php?ID=179694

Original article

MODIFICATION OF THE ASYNCHRONOUS ANT COLONY METHOD FOR SEARCHING FOR RATIONAL SOLUTIONS TO A PARAMETRIC PROBLEM

Yuri P. Titov[1]✉, Vladimir A. Sudakov[2]

[1]Moscow Aviation Institute (National Research University), Moscow, Russia

[2]Keldysh Institute of Applied Mathematics, KIAM, Moscow, Russia

[1]kalengul@mail.ru✉

[2]sudakov@ws-dss.com

Abstract. The article considers the ant colony method modifications for application in the problems of searching for optimal values of system parameters. The parameters optimality

is being determined by the values of the objective function computed as the result of complex computations of a mathematical or simulation model running on a separate computing cluster with the ability for parallel computing of the objective function values. To save the computing cluster resources, the already considered parameter values and the value of the objective function are stored in hash tables. Modifications of the ant colony method are being developed to solve the problem of stagnation, convergence to one rational solution, and allow the method to be applied to find the optimal solution without using the multi-start procedure. This is achieved by modifying the behavior of the agent ant; the ants continue to search for a new set of parameter values that have not yet been considered on the cluster. Thus, each ant agent searches for its own unique route. As the result, the modification of the ant colony method does not converge to one solution, but continues to search for other solutions, avoiding stagnation and ensuring a complete directed search of all solutions (values of system parameters). This modification determines optimal and rational solutions in the early (estimate of mathematical expectation) iterations of the method.

The authors proposed an asynchronous modification of the ant colony method for the effective work with a remote multi-threaded computer. Each ant agent searches for its unique set of parameter values in a separate thread and, after finding it, sends a new task to the computing cluster via TCP sockets. The tasks received on the computer are buffered and sent to the dispatcher for uniform and constant loading of the computational threads. of The ant agents blocking associated with the addition and evaporation of pheromone is solved by the blocking-free RCU (read, copy, update) algorithm. A separate thread

controls creation of a new graph with an updated pheromone, and all new agent ants will look for paths in the already new graph.

Keywords: ant colony method, metaheuristic optimization, directed search, parametric problem, computing cluster, algorithms that do not use locks

For citation: Titov Yu.P., Sudakov V.A. Modification of the asynchronous ant colony method for searching for rational solutions to a parametric problem. *Trudy MAI*, 2024, no. 135. URL: https://trudymai.ru/eng/published.php?ID=179694

Введение

Задачи параметрической оптимизации возникают при необходимости определения набора параметров некоторой системы с целью оптимизации некоторого критерия, в общем случае векторного. Среди данного класса задач, как пример, можно рассмотреть задачу расположения блоков, узлов и агрегатов на корпусе космического корабля, отсека, спутника. Целевой функцией такой задачи является определение центра масс и его отклонения от оси. Возможно использование и векторного критерия, учитывающего доступность приборов, удобство и длину кабелей и трубопроводов, различное взаимовлияние близко расположенных блоков и т.д. Решением такой задачи является набор координат размещения (в работе рассматривается левый верхний угол) блока с определенными массогабаритными параметрами, коннекторами и портами [1]. Данная задача относится к классу задач непрерывного нелинейного программирования, которая в результате дискретизации решений с определенной точностью может быть сведена к задаче дискретного программирования. Подобная дискретизация может возникать

из-за особенностей расположения технологических крепежных отверстий на корпусе космического аппарата. В этом случае значениями параметров будут координаты крепежных отверстий, или их номер.

Подобные задачи называют задачами параметрической оптимизации – поиска оптимальных значений параметров. Такие задачи позволяют решать не только задачи анализа системы, но и задачи синтеза [2-4]. При поиске оптимальных дискретных значений параметров системы применяют методы неявного перебора, например, метод неявного перебора по векторной решетке, а при наложении условия сепарабельности функции – методы динамического программирования [5-7]. Данные методы позволяют находить оптимальное значение целевой функции, но чувствительны к виду функции и размерности решаемой задачи. Метода неявного перебора по векторной решетке чувствителен к порядку рассмотрения вершин при использовании критерия планомерного исключения альтернатив [8]. Аналогично различную эффективность работы при различном порядке рассмотрения параметров показывает и алгоритм динамической оптимизации.

Параметрическая оптимизация обычно осуществляется методами оптимизации, сходящимися к одному оптимальному значению, например, методами Хука-Дживса, Нелдера-Мида или градиентным спуском [8]. Такие методы хорошо работают с однокритериальными одноэкстремальными задачами. Для многокритериальных задач применяют различные виды свертки к одному критерию. Для многоэкстремальных задач (также овражных) применяют процедуру мультистарта, повторного многократного запуска оптимизации из различных начальных условий [9-11]. Такой подход не позволяет доказать точное нахождение

оптимального решения и наилучшее из найденных решений называется рациональным. Рациональных решений может быть много и, в случае многокритериальной оптимизации, векторным критерием, позволяет применять системы поддержки принятия решений, взаимодействие с пользователем. В случае наличия нескольких оптимальных решений, с одинаковыми значениями лицо, принимающее решение может выбрать одно из них, так как построенная оптимизационная модель не полностью отражает реальную ситуацию.

Другим способом поиска оптимального решения является полный перебор всех комбинаций параметров. Полный перебор гарантирует нахождение оптимального и всех рациональных наборов значений параметров. Для сокращения количества рассматриваемых наборов значений применяют направленный перебор, например, метод неявного перебора по векторной решетке [8].

В работе рассматривается применение и модификации метода муравьиных колоний для решения параметрической задачи [11-16]. Предложенный в 1992 году алгоритм сходятся к одному решению, когда муравьи на каждой итерации выбирают один и тот же путь в графе. Предложенные в работе модификации позволяют методу не сходиться к одному решению, что свойственно как методу оптимизации, а продолжать поиск решений, осуществив полный перебор всех вариантов при отсутствии остановки алгоритма. Из-за случайной природы поиска маршрута в методе муравьиных колоний для исследования проводилось множество прогонов, с последующем сбором статистической информации, оценками математического ожидания и доверительными интервалами. Тестирование эффективности применения алгоритмов проводилось как на двумерных бенчмарках,

многоэкстремальных и овражных функции [17-19], так и на больших параметрических графах [20].

Параллельный поиск путей муравьями позволяет обеспечить параллельные вычисления при поиске рационального решения, применяя сложные вычислительные кластеры и параллельные вычисления. В работе предложена структура программного обеспечения, позволяющая оптимизировать нагрузку на вычислитель, уменьшив время его простоя. Такой подход позволяет применять распределенные выделенные сервера для ускорения направленного перебора значений параметров системы.

Модификации метода муравьиных колоний для решения параметрической задачи

Решение параметрической задачи требует сложных вычислений, которые эффективно выполнять на кластерных системах. Критерием эффективного использования кластера является его загрузка, поэтому повторно найденные решения нецелесообразно отправлять на вычислительный кластер. Метод муравьиных колоний сходится к определенному одному решению тогда, когда муравьи выбирают одинаковый путь в графе [12]. Для решения проблемы вычисления значений критерия при повторном выборе уже рассмотренного маршрута используется дополнительная структура данных, хэш-таблица. Перед отправкой на вычислительный кластер, проверяется наличие решения в хэш-таблице и, если такое решение не найдено, то решение считается новым и для него определяется значение критерия на кластере. Если решение найдено в хэш-таблице, то значение критерия может быть определено в таблице без необходимости

проведения расчетов на кластере. Время поиска значений в хэш-таблице зависит от количества рассмотренных решений, так как увеличивается вероятность коллизии.

Для решения параметрической задачи методом муравьиных колоний применяется структура данных, в которой методом муравьиных колоний выбирается значение каждого параметра. В данной работе рассматривается дискретная оптимизация, в которой для каждого параметра системы определяется конечный набор дискретных значений [12-13]. Муравей последовательно для каждого параметра определяет одно значение (вершину графа) исходя из вероятностного выбора. Полученный набор значений для каждого параметра (путь муравья в виде последовательности вершин) определяет решение.

Вероятностная формула выбора вершины требует изменения, в оригинальной формуле используется априорная информация о длине дуги (оригинальный алгоритм решает задачу коммивояжера), которая недоступна для параметрической задачи [12]. Если выбирать вершину только исходя из количества феромона в вершине, то алгоритм стагнирует, сходится к первому хорошему решению. Для решения проблемы стагнации, как и в случае оптимизационных алгоритмов, применяют мультистарт [9-10].

Для решения проблемы стагнации предложена модификация вероятностной формулы, учитывающая не только количество феромона, но и количество посещений агентами вершины (1). За счет аддитивной свертки происходит автокомпенсация слагаемых. Такой подход позволяет алгоритму на начальном этапе поиска искать новые решения, в последствии переходя к поиску наилучшего решения с точки зрения значений целевой функции [19-20].

$$P_{j,k}(t) = \frac{k1*\mu(t)^{\alpha}_{norm,j} + k2*(1/kol(t)_j)^{\beta} + k3*(kol(t)_j/MaxKol_j)^{\gamma}}{\sum_{z \in J_k}(k1*\mu(t)^{\alpha}_{norm,j} + k2*(1/kol(t)_j)^{\beta} + k3*(kol(t)_j/MaxKol_j)^{\gamma})} \quad (1)$$

где, $P_{j,k}(t)$ – вероятность выбора j-ой вершины слоя k-м муравьем на итерации t; J_k - множество вершин, в которые может перейти муравей k из текущей вершины; $k1, k2, k3, \alpha, \beta, \gamma$ – коэффициенты; $\mu(t)^{\alpha}_{norm,j}$ – нормированное количество феромона в вершине j на итерации t; $kol(t)_j$ – количество посещений муравьями вершины j на итерации t; $MaxKol_j$ – максимальное количество решений, в которых может присутствовать вершина j.

Применение новой вероятностной формулы (1) уменьшает проблему стагнации, алгоритм сходится к одному решению и вероятность выбора альтернативных решений стремиться к 0. Дальнейший поиск возможен, если каждое решение будет заносить феромон только определенное количество раз, например, только при первом нахождении. При наличии хранилища решений в хэш-таблице определить повторное нахождение решения не составляет труда. В работах [19-22] рассматривались следующие предложенные модификации алгоритма:

- ACOCN (ACO Cluster New) классический метод муравьиных колоний с применением хэш-таблицы и получением значений целевой функции без обращения к вычислителю (вычислительному кластеру).
- ACOCNI (ACO Cluster New Ignor) если муравей нашел уже рассмотренное решение, то данный муравей не заносит феромон на параметрический граф – игнорируется. В алгоритме каждый путь один раз учитывается в графе.

- ACOCCyN (ACO Cluster Cycle N) если муравей нашел уже рассмотренное решение, то производит дальнейший циклический поиск нового решения. Цикл ограничен N итераций, если новое решение не найдено – то муравей игнорируется.
- ACOCCyI (ACO Cluster Cycle Infinity) если муравей нашел уже рассмотренное решение, то производит дальнейший циклический поиск пока не будет найдено новое решение.
- ACOCT (ACO Cluster Tree) Если муравей нашел уже рассмотренное решение, то новое решение ищется алгоритмом неявного перебора, рассматривая параметрический граф как дерево.

Исследование модификаций метода муравьиных колоний

Для проведения исследования рассматривались бенчмарки: «Carrom table function», функция Розенброка, синусоидальная функция Швефеля, функция Шаффера, функция Растригина и других бенчмарков из работ [17-19].

При проведении множества прогонов оценивались:

- Вероятность нахождения и номер итерации, на которой на кластер было отправлено оптимальное решение. Данное исследование позволяет судить об эффективности работы модификаций метода муравьиных колоний. Если оптимальных решений несколько, то оценивается вероятность поиска всех решений.
- Количество успешных обращений к хэш-таблице. Данный параметр позволяет оценить вероятность нахождения агентом нового решения.

На основе данных показателей имеется возможность сравнивать полученные модификации, как с методами оптимизации, так и с алгоритмом полного перебора. Каждое обращение к хэш-таблице вызывает простой вычислительного кластера, при полном переборе решений каждое решение рассматривается только один раз. Но для полного перебора, без процедуры неявного перебора или переупорядочивания решений, находит решение в среднем после рассмотрения 50% всех решений, т.е. вероятность выбора любого решения одинакова и не изменяется. С другой стороны, оптимизационные алгоритмы не осуществляет полный перебор вариантов, и требуют мультистарт для нахождения всех оптимальных решений.

Модификация ACOCN стагнирует и оптимальное решение без мультистарта только в 20% запусков находит оптимальное значение целевой функции (рис. 1). При этом количество итераций, необходимых для поиска оптимальных решений, в среднем в 5 раз меньше, чем при использовании модифицированных алгоритмов [19].

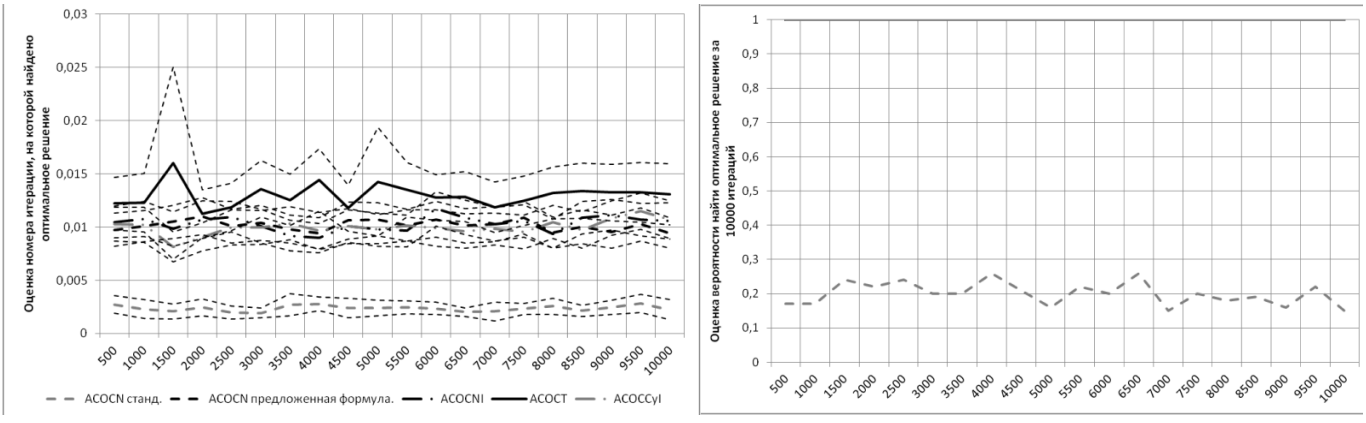
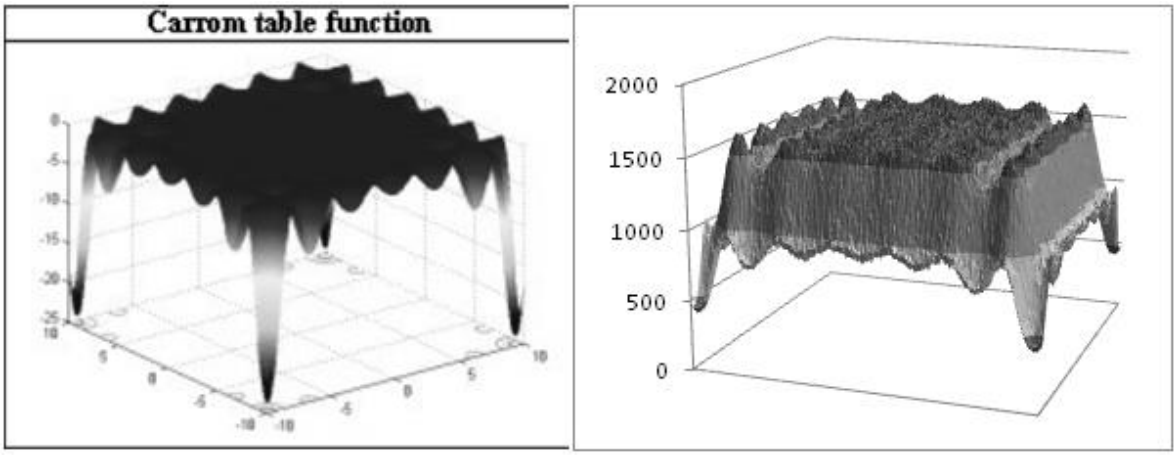


Рис 1. Оценка эффективности работы предложенных модификаций метода муравьиных колоний

Модификации, основанные на повторном поиске нового решения методом муравьиных колоний (ACOCCyN, ACOCCyI), показывают наилучшую эффективность [19]. В основе данных алгоритмов лежит подход, позволяющий каждому муравью искать новое решение повторным запуском метода муравьиных колоний, т.е. каждый муравей находит новое решение. Данные модификации позволяют, для дискретной задачи, точно определить требуемое количество муравьев.

По результатам исследований данные модификации позволяют рассмотреть все рациональные и найти все оптимальные решения на ранних итерациях (до 5% от всех рассмотренных решений) [19-20]. На рисунке 2 приведены графики функций бенчмарков (слева) и оценка математического ожидания номера итерации на котором найдено данное решение (справа).



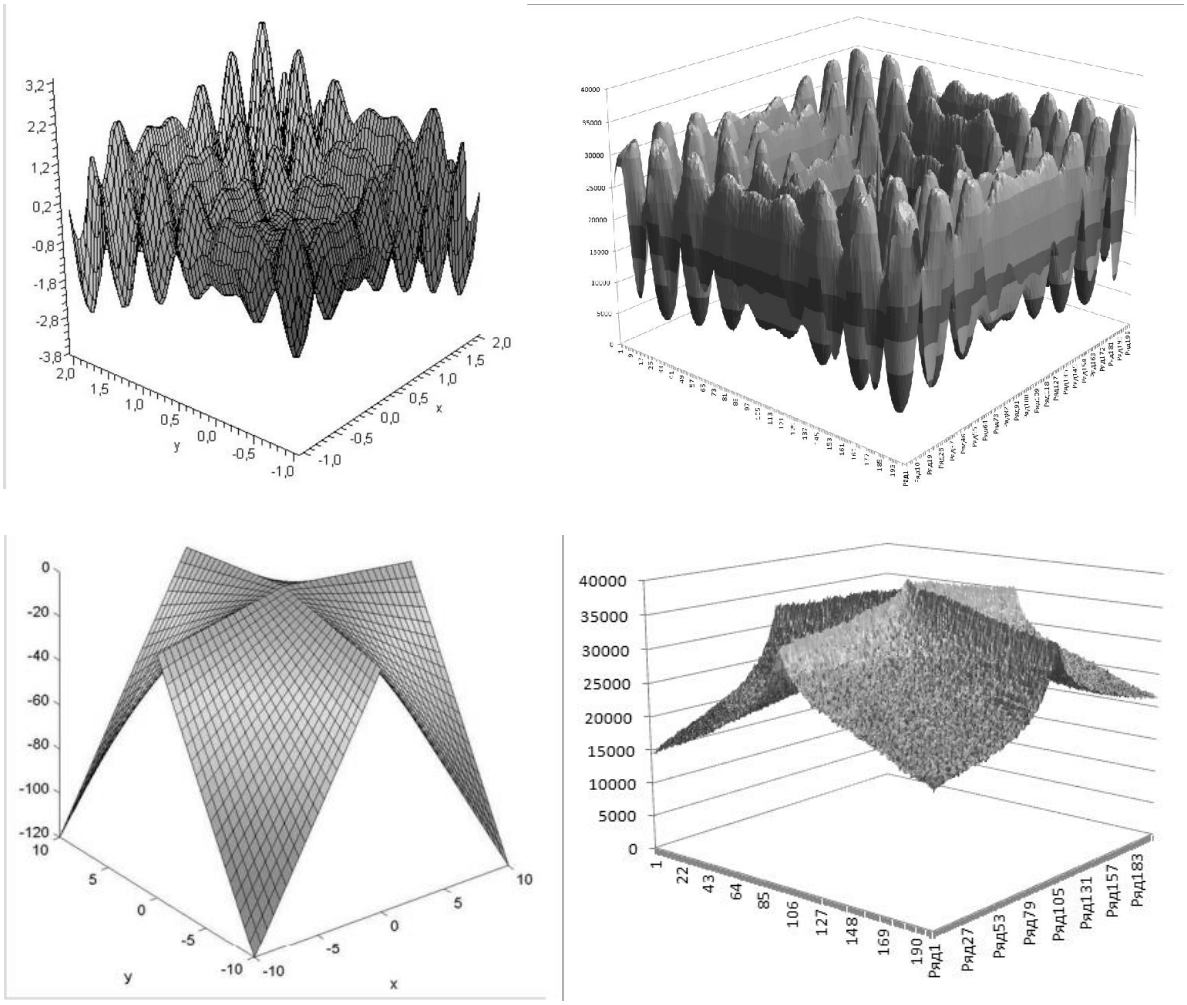


Рис 2. График функции бенчмарка «Carrom table function» (сверху), «Мульти-функции» (посередине) и функции «Шеффеля» (снизу). Слева график функции, справа оценка математического ожидания номера решения.

Асинхронные модификации метода муравьиных колоний.

Проведенные исследования показали эффективность применения новой вероятностной формулы и модификаций алгоритма для решения параметрической задачи. В рамках данной задачи предполагается не сходимость и поиск одного оптимального решения, а предложение пользователю рациональных решений. Пользователь самостоятельно определяет момент остановки работы алгоритма. Если пользователь не остановит работу алгоритма, то будут рассмотрены все возможные решения (в случае дискретной, ограниченной задачи). Для эффективного

взаимодействия с пользователем разработан web-интерфейс, работающий в отдельном потоке и выводящий пользователю информацию о текущем оптимальном решении и проценте рассмотренных решений.

Время работы вычислительного кластера является основным вычислительным ресурсом для подобной системы. Чем раньше будет обнаружено решение, удовлетворяющее пользователя (рациональное или оптимальное решение), тем раньше будет остановлена работа программы и меньше загружен вычислительный кластер. Другим способом повышения эффективности работы кластера является уменьшение времени простоя кластера за счет разработки асинхронного метода муравьиных колоний.

Алгоритм муравьиных колоний (рис 3) состоит из следующих этапов:

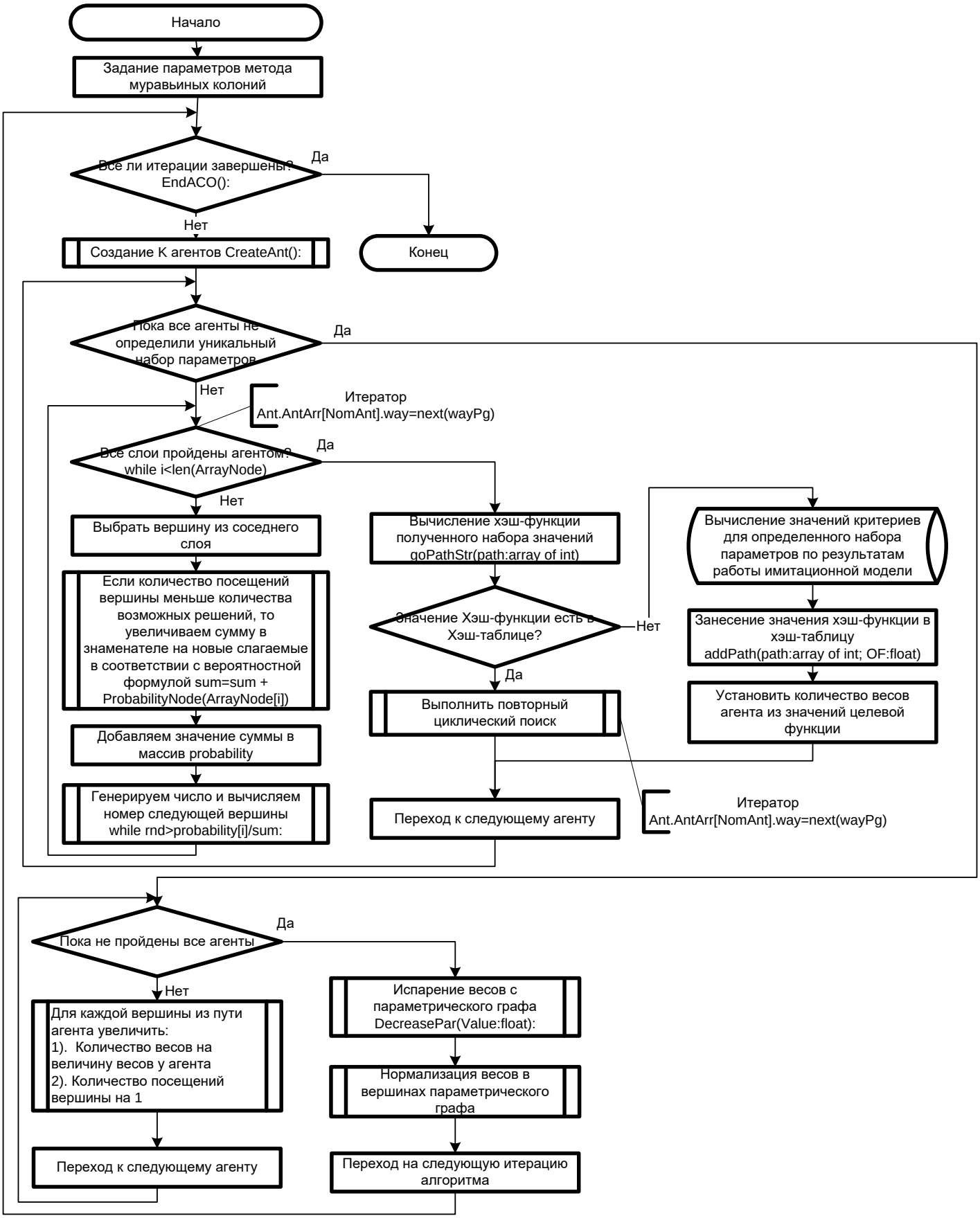


Рис 3. Алгоритм работы модификаций метода муравьиных колоний

(синхронный вариант)

1. Создается группа муравьев. Размер группы задается константой N.
2. В цикле для каждого муравья (агента, с точки зрения агентного моделирования):
    a. Определяется путь в графе. В случае параметрической задачи для каждого муравья определяется набор значений параметров.
    b. Поиск наличия решения в структуре хранения – хэш-таблице
    c. Если решение в хэш-таблице найдено, то возврат к пункту a).
    d. Отправка решения на кластер и вычисление значения целевой функции. Взаимодействие с кластером осуществляется через сокетное взаимодействие. Кластер имеет сокет-сервер, к которому подключаются сокет-клиенты из программы, реализующей метод муравьиных колоний. Значение целевой функции, вычисленной на кластере, отправляется через сокетное соединение.
    e. Полученное решение и значение целевой функции заносится в хэш-таблицу.
3. Происходит обновление состояния параметрического графа (состояние среды):
    a. Испарение феромона, уменьшение количества феромона во всех вершинах, значениях параметров
    b. Добавление феромона в зависимости от найденных решений муравьев на итерации. Для каждой вершины из пути муравья (значения параметров решения муравья) добавляется

дополнительный феромон и увеличивается количество посещений вершины.

4. Завершение итерации, проверка условия окончания и переход к следующей итерации. Переход к шагу 1.

В предложенном алгоритме муравьи-агенты последовательно выполняют однотипные действия (цикл 2) при неизменном состоянии параметрического графа. Параметрический граф изменяет свое состояние только в пункте 3. Если для каждого муравья-агента создать свой поток и выполнять этап 2 асинхронно другим муравьям-агентам. При этом взаимодействие с параметрическим графом осуществляется только в режиме чтения, поэтому блокировки агентов не происходит. Основными критическими секциями являются: взаимодействие с кластером и добавление в хэш-таблицу.

Сокет-взаимодействие с кластером позволяет организовать критическую секцию и блокировать агента до тех пор, пока кластер не пришлет значение целевой функции. Для работы с хэш-таблицей необходимо создать критические секции на основе мьютексов чтения-записи. Так как запись в хэш-таблицу происходит после получения значения целевой функции от кластера, то можно осуществить запись в хэш-таблицу в одной блокировке взаимодействия с кластером. Но при добавлении значений в хэш-таблицу ее необходимо заблокировать по записи и запретить другим агентам читать из нее.

Подобный асинхронный алгоритм позволяет вычислять пути агентов на отдельных ядрах (процессорах), кластер при этом ожидает, когда ему будут переданы уникальные пути для каждого муравья-агента. Так как одновременно

несколько агентов будут по сокетным соединениям отправлять свои решения, то кластер будет постоянно загружен вычислениями. При этом диспетчеризация вычислений значений целевой функции осуществляется на кластере путем передачи сообщений потокам вычислителя. Применение сокетных соединений, хоть и накладывает дополнительные издержки на создание TCP-соединения, но при этом позволяет не использовать общую критическую секцию для каждого агента. Компромиссным решением является создание ограниченного количества TCP-соединений (по количеству муравьев на итерации) и проведение трехстороннего рукопожатия по всем соединениям до начала работы алгоритма. После получения значения целевой функции TCP-соединение не закрывается, а освобождается и передается следующему созданному агенту.

По окончанию нахождения маршрута агенты ожидают изменения состояния графа, испарения и занесения феромона для значений параметров. Именно последовательное изменение состояния графа позволяет осуществлять направленный перебор, и, в оригинальном алгоритме, изменение состояния производится после вычисления определенного количества решений (задается параметром алгоритма N). При изменении графа поиск решений муравьями-агентами невозможен и в это время вычислительный кластер будет простаивать.

Для решения проблемы блокировки агентов применяется свободный от блокировки алгоритм RCU (прочитать, скопировать, обновить) [23]. Обновление состояния графа происходит в отдельном потоке, который большую часть времени заблокирован и разблокируется только когда определенное количество муравьев вычислят свои маршруты. Данный поток копирует состояние в новую структуру

данных, обращаясь к старой только для чтения без блокирования агентов. Испарение и добавление феромона происходит в новой структуре. После завершения всех операций новая структура данных подается как состояние системы для новых агентов. Старые агенты, которые в данный момент ищут решение, определяют значения параметров, выполняют вероятностный поиск в старой структуре со старым состоянием системы. В результате в методе муравьиных колоний одновременно имеется два графа одинаковой структуры, но разного состояния, в процессе добавления и испарения феромона муравьи-агенты не блокируются и продолжают поиск в старой структуре. После того, как все агенты закончили работу со старой структурой, она удаляется из системы.

Общий вид асинхронного метода муравьиных колоний приведен на рисунке 4. Асинхронная модификация метода муравьиных колоний требует дополнительных параметров, кроме «стандартных» параметров алгоритма – коэффициента испарения и коэффициента добавления феромона, добавляются параметры новой вероятностной формулы – коэффициенты и степени при слагаемых. Также требуется определить граничное значение муравьев-агентов: количество активных муравьев-агентов, количество муравьев-агентов при котором происходит обновление графа. Если второй параметр схож по своей природе с количеством агентов на итерации, то количество активных муравьев-агентов определяется количеством параллельных потоков. Так как вычисления пути муравьями-агентами требуют минимальные задержки на блокировки, то их можно считать CPUBound потоками, следовательно, количество активных муравьев-агентов определяется числом ядер вычислительной системы.

Отдельно стоит отметить, что работа потока ММК и основного потока не является CPUBound процессами, так как большую часть времени не выполняют полезных действий, а находятся в состоянии блокировки. Основной поток разблокирует завершение работы потока муравья-агента, а поток ММК будет разблокирован при определенных условиях основным потоком. После выполнения необходимых действий потоки уходят в состояние блокировки и не потребляют процессорное время, на данные потоки можно выделить одно ядро.

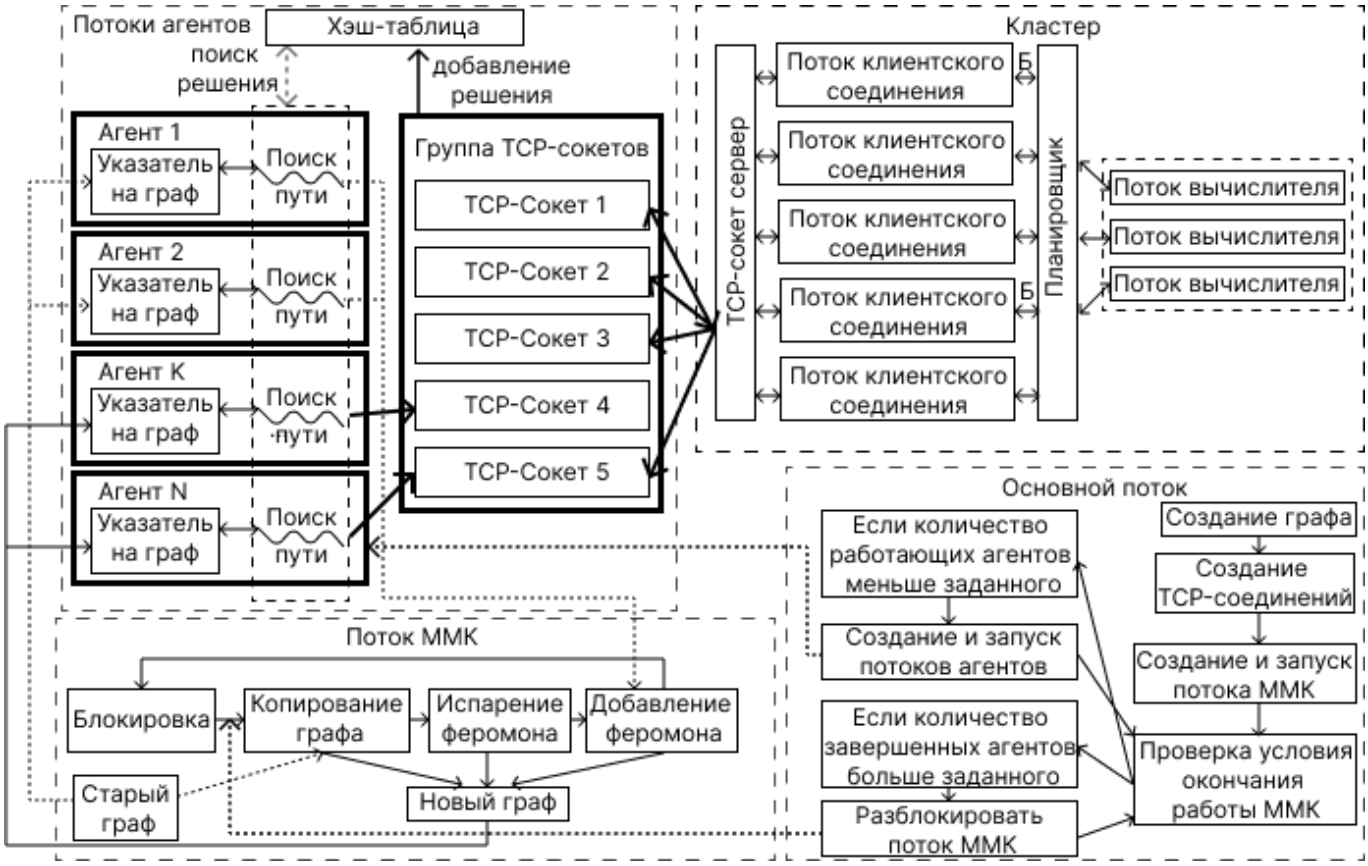


Рис 4. Схема работы блоков алгоритма асинхронного метода муравьиных колоний.

Вычислительный кластер располагается на другой вычислительной машине, и передача информации происходит по сети по результатам сокетного взаимодействия. Количество потоков вычислителя определяется количество доступных ядер на машине вычислителя, при этом количество клиентских

подключенных сокетов должно быть больше или равно количеству потоков вычислителя.

Обсуждение и выводы

В работе рассмотрен нестандартный подход к решению параметрической задачи. Метод метаэвристической оптимизации – метод муравьиных колоний, который обеспечивает сходимость к одному решению, модифицирован для осуществления полного перебора всех решений. В результате предложенный метод выполняет полный перебор всех вариантов дольше, чем метод полного перебора так как некоторые решения будут найдены повторно, но при этом предложенные модификации проигрывают методам оптимизации по скорости нахождения оптимального решения.

Предложенные модификации эффективно работают при взаимодействии с пользователем. В задачах, которые имеют несколько оптимальных решений, с одинаковыми значениями целевой функции, но различными для пользователя или задачах, где пользователю, в силу неточности построенной модели или многоэкстремальной оптимизации, требуется нахождение рационального решения, предложенные модификации максимально эффективны. Хотя первое оптимальное решение предложенные модификации метода муравьиных колоний находят на более поздних итерациях, чем методы оптимизации, но все остальные оптимальные и рациональные решения модификации находят на ранних итерациях без использования мультистарта. По оценкам математического ожидания номера итерации предложенные модификации не чувствительны к сложным овражным и многоэкстремальным функциям.

Предложенный асинхронный метод муравьиных колоний позволяет производить параллельные оптимизационные вычисления. Если в предложенной модификации задать всего один активный поток для муравья-агента, то эффективность его работы будет близкой к работе последовательного, синхронного метода. Параллельный поиск решений позволяет эффективно решать оптимизационную задачу на кластерных системах с множеством потоков.

Выполнение асинхронной работы агентов по поиску пути проводилось исследователями, но из-за необходимости ожидания всех агентов из группы для завершения итерации не приводит к существенному увеличению эффективности алгоритма. Предложенный асинхронная модификация метода муравьиных колоний направлена на повышение эффективности использования вычислителя, уменьшения времени простоя устройства. Это обеспечивается применением алгоритмами, свободными от блокировок и вынесением процесса изменения состояния графа (добавление и испарение феромона) в отдельный поток.

Исследование проводилось на распространенных двумерных бенчмарках и на тестовых графах большой размерности. Практическое применение и исследование на реальных примерах затруднено, так как модель размещения блоков космического аппарата находится на стадии разработки. Однако следует отметить, что проведенные исследования позволяют применит новые эффективные модификации методы муравьиных колоний с рациональными наборами параметров.

Список источников

1. Пантелеев А.В., Дмитраков А.В. Применение метода дифференциальной эволюции для оптимизации параметров аэрокосмических систем // Труды МАИ. 2010. № 37. URL: https://trudymai.ru/published.php?ID=13428

2. Пантелеев А.В., Летова Т.А., Помазуева Е.А. Применение методов глобальной оптимизации для параметрического синтеза обобщенного пропорционально-интегрально-дифференциального регулятора в задаче управления полетом // Труды МАИ. 2015. № 79. URL: https://trudymai.ru/published.php?ID=55635

3. Пантелеев А.В., Каранэ М.С. Применение гибридного мультиагентного метода интерполяционного поиска в задаче о стабилизации спутника // Труды МАИ. 2021. № 117. URL: https://trudymai.ru/published.php?ID=156249. DOI: 10.34759/trd-2021-117-10.

4. Пантелеев А.В., Алешина Е.А. Применение метода частиц в стае к задаче поиска оптимального управления дискретными детерминированными системами // Труды МАИ. 2010. № 37. URL: https://trudymai.ru/published.php?ID=13427

5. Таха Х. Введение в исследование операций. - М.: Издательский дом "Вильямс", 2005. – 912 с.

6. Хахулин Г.Ф. Постановки и методы решения задач дискретного программирования. - М.: Изд-во МАИ, 1992 – 59 с.

7. Пападимитриу Х., Стайглиц К. Комбинаторная оптимизация. Алгоритмы и сложность. - М.: Мир, 1985. – 512 с.

8. Хахулин Г.Ф., Красовская М.А., Булыгин В.С. Теоретические основы автоматизированного управления (Задачи, методы, алгоритмы теории оптимального планирования и управления). - М.: Изд-во МАИ, 2005. – 396 с.

9. Карпенко А.П. Современные алгоритмы поисковой оптимизации. Алгоритмы, вдохновленные природой. - М.: Изд-во МГТУ им. Баумана, 2017. - 446 с.

10. Саймон Д. Алгоритмы эволюционной оптимизации: практическое руководство. - М.: ДМК Пресс, 2020. - 1002 с.

11. Юхименко Б.И., Титов Н.А., Ушаков В.О. Разработка и исследование алгоритмов муравьиной колонии для решения некоторых задач комбинаторной оптимизации // Актуальные научные исследования в современном мире. 2020. № 11-2 (67). С. 101-115.

12. Dorigo M., Stutzle T. Ant Colony Optimization, MIT Press, 2004, 321 p.

13. Bergstra James S., Rémi Bardenet, Yoshua Bengio, Balázs Kégl. Algorithms for hyper-parameter optimization // In Advances in neural information processing systems, 2011, pp. 2546-2554.

14. Joseph M. Pasia, Richard F. Hartl, Karl F. Doerner. Solving a Bi-objective Flowshop Scheduling Problem by Pareto-Ant Colony Optimization // Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics, International Workshop, 2006, pp. 294–305. DOI: 10.1007/978-3-540-74446-7_15

15. Parpinelli R., Lopes H., Freitas A. Data mining with an ant colony optimization algorithm // IEEE Transactions on Evolutionary Computation, 2002, vol. 6 (4), pp. 321–332. DOI: 10.1109/TEVC.2002.802452

16. Martens D., De Backer M., Haesen R., Vanthienen J., Snoeck M., Baesens B. Classification with ant colony optimization // IEEE Transactions on Evolutionary Computation, 2007, vol. 11 (5), pp 651–665. DOI: 10.1109/TEVC.2006.890229

17. Mishra Sudhanshu K. Some New Test Functions for Global Optimization and Performance of Repulsive Particle Swarm Method // SSRN Electronic Journal, 2006. DOI: 10.2139/ssrn.926132

18. Layeb Abdesslem. New hard benchmark functions for global optimization, 2022. DOI: 10.48550/arXiv.2202.04606

19. Sinitsyn I.N., Titov Y.P. Control of Set of System Parameter Values by the Ant Colony Method // Automation and Remote Control, 2023, vol. 84, pp. 893–903. DOI: 10.1134/S0005117923080106

20. Судаков В.А., Титов Ю.П., Сивакова Т.В., Иванова П.М. Применение метода муравьиных колоний для поиска рациональных значений параметров технической системы // Препринт ИПМ. 2023. № 38. С. 1-15. DOI: 10.20948/prepr-2023-38

21. Синицын И.Н., Титов Ю.П. Исследование алгоритмов циклического поиска дополнительных решений при оптимизации порядка следования гиперпараметров методом муравьиных колоний // Системы высокой доступности. 2023. Т. 19. № 1. С. 59-73. DOI: 10.18127/j20729472-202301-05

22. Синицын И.Н., Титов Ю.П. Исследование возможности получения всех решений методом муравьиных колоний для задачи оптимизации порядка следования гиперпараметров // Системы высокой доступности. 2023. Т. 20. № 2. С. 55-69. DOI: 10.31857/S000523102308010X

23. Paul E. McKenney. Comparing performance of read-copy update and other locking primitives // Sequent TR-SQNT-98-PEM-1, 1998.

References

1. Panteleev A.V., Dmitrakov A.V. *Trudy MAI*. 2010, no. 37. URL: https://trudymai.ru/eng/published.php?ID=13428

2. Panteleev A.V., Letova T.A., Pomazueva E.A. *Trudy MAI*, 2015, no. 79. URL: https://trudymai.ru/eng/published.php?ID=55635

3. Panteleev A.V., Karane M.S. *Trudy MAI*, 2021, no. 117. URL: https://trudymai.ru/eng/published.php?ID=156249. DOI: 10.34759/trd-2021-117-10

4. Panteleev A.V., Aleshina E.A. *Trudy MAI*, 2010, no. 37. URL: https://trudymai.ru/eng/published.php?ID=13427

5. Takha Kh. *Vvedenie v issledovanie operatsii* (Introduction to Operations Research), Moscow, Izdatel'skii dom "Vil'yams", 2005, 912 p.

6. Khakhulin G.F. *Postanovki i metody resheniya zadach diskretnogo programmirovaniya* (Statements and methods for solving discrete programming problems), Moscow, Izd-vo MAI, 1992, 59 p.

7. Papadimitriu Kh., Staiglits K. *Kombinatornaya optimizatsiya. Algoritmy i slozhnost'* (Combinatorial optimization. Algorithms and complexity), Moscow, Mir, 1985, 512 p.

8. Khakhulin G.F., Krasovskaya M.A., Bulygin V.S. *Teoreticheskie osnovy avtomatizirovannogo upravleniya. Zadachi, metody, algoritmy teorii optimal'nogo planirovaniya i upravleniya*. (Theoretical foundations of automated control. Tasks,

methods, algorithms of the theory of optimal planning and control), Moscow, Izd-vo MAI, 2005, 396 p.

9. Karpenko A.P. *Sovremennye algoritmy poiskovoi optimizatsii. Algoritmy, vdokhnovlennye prirodoi* (Modern search engine optimization algorithms. Algorithms inspired by nature), Moscow, Izd-vo MGTU im. Baumana, 2017, 446 p.

10. Saimon D. *Algoritmy evolyutsionnoi optimizatsii: prakticheskoe rukovodstvo* (Evolutionary optimization algorithms: a practical guide), Moscow, DMK Press, 2020, 1002 p.

11. Yukhimenko B.I., Titov N.A., Ushakov V.O. *Aktual'nye nauchnye issledovaniya v sovremennom mire*, 2020, no. 11-2 (67), pp. 101-115.

12. Dorigo M., Stutzle T. *Ant Colony Optimization*, MIT Press, 2004, 321 p.

13. Bergstra James S., Rémi Bardenet, Yoshua Bengio, Balázs Kégl. Algorithms for hyper-parameter optimization, *In Advances in neural information processing systems*, 2011, pp. 2546-2554.

14. Joseph M. Pasia, Richard F. Hartl, Karl F. Doerner. Solving a Bi-objective Flowshop Scheduling Problem by Pareto-Ant Colony Optimization, *Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics*, International Workshop, 2006, pp. 294–305. DOI: 10.1007/978-3-540-74446-7_15

15. Parpinelli R., Lopes H., Freitas A. Data mining with an ant colony optimization algorithm, *IEEE Transactions on Evolutionary Computation*, 2002, vol. 6 (4), pp. 321–332. DOI: 10.1109/TEVC.2002.802452

16. Martens D., De Backer M., Haesen R., Vanthienen J., Snoeck M., Baesens B. Classification with ant colony optimization, *IEEE Transactions on Evolutionary Computation*, 2007, vol. 11 (5), pp 651–665. DOI: 10.1109/TEVC.2006.890229

17. Mishra Sudhanshu K. Some New Test Functions for Global Optimization and Performance of Repulsive Particle Swarm Method, *SSRN Electronic Journal*, 2006. DOI: 10.2139/ssrn.926132

18. Layeb Abdesslem. *New hard benchmark functions for global optimization*, 2022. DOI: 10.48550/arXiv.2202.04606

19. Sinitsyn I.N., Titov Y.P. Control of Set of System Parameter Values by the Ant Colony Method, *Automation and Remote Control*, 2023, vol. 84, pp. 893–903. DOI: 10.1134/S0005117923080106

20. Sudakov V.A., Titov Yu.P., Sivakova T.V., Ivanova P.M. *Preprint IPM,* 2023, no. 38, pp. 1-15. DOI: 10.20948/prepr-2023-38

21. Sinitsyn I.N., Titov Yu.P. *Sistemy vysokoi dostupnosti*, 2023, vol. 19, no. 1, pp. 59-73. DOI: 10.18127/j20729472-202301-05

22. Sinitsyn I.N., Titov Yu.P. *Sistemy vysokoi dostupnosti*, 2023, vol. 20, no. 2, pp. 55-69. DOI: 10.31857/S000523102308010X

23. Paul E. McKenney. Comparing performance of read-copy update and other locking primitives, *Sequent TR-SQNT-98-PEM-1*, 1998.

Статья поступила в редакцию 06.03.2024
Одобрена после рецензирования 10.03.2024

Принята к публикации 26.04.2024

The article was submitted on 06.03.2024; approved after reviewing on 10.03.2024; accepted for publication on 26.04.2024