ISSN 2782-2001
Системы анализа
и обработки данных
том 103, № 3, 2026, с. 97–114

http://journals.nstu.ru/vestnik
Analysis and data
processing systems
Vol. 103, No. 3, 2026, pp. 97–114

ИНФОРМАЦИОННЫЕ
ТЕХНОЛОГИИ
И ТЕЛЕКОММУНИКАЦИИ

INFORMATION
TECHNOLOGIES
AND TELECOMMUNICATIONS

УДК 519.876

DOI: 10.17212/2782-2001-2026-3-97-114

Применение алгоритма муравьиной колонии в оптимизации управления мультипроектными разработками*

Ю.А. МЕЗЕНЦЕВ[a], А.А. БОРИСОВА[b]

РФ, 630073, г. Новосибирск, пр. Карла Маркса, 20, Новосибирский государственный технический университет

[a] *mesyan@yandex.ru* [b] *alina_10112004@mail.ru*

В статье рассматривается применение муравьиного алгоритма (Ant Colony Optimization, ACO) для решения задачи оптимизации управления мультипроектными разработками с ограниченными ресурсами. Мультипроектная среда характеризуется конкуренцией между задачами за ресурсы, технологическими и ресурсными зависимостями, что делает задачу комбинаторной и NP-трудной.

В работе описан алгоритм муравьиной колонии, включающий формирование допустимых расписаний одновременно несколькими искусственными муравьями, вероятностный выбор операций с учетом уровня феромона и эвристической информации, а также обновление феромонных следов в процессе итераций. Проведены эксперименты на задачах различной размерности, показавшие, что при корректной настройке параметров алгоритм стабильно находит оптимальные или близкие к оптимальным решения. Для малых задач решение полностью совпадает с результатами точной оптимизации в IBM ILOG CPLEX, для средних и больших задач метод обеспечивает эффективное нахождение качественных расписаний за существенно меньшее время по сравнению с точными методами.

Результаты исследования демонстрируют полиномиальную вычислительную трудоемкость алгоритма ACO и его высокую эффективность при планировании мультипроектных разработок, что делает его перспективным инструментом для применения в практических задачах управления ресурсами и сроками проектов.

Ключевые слова: мультипроектное планирование, комбинаторная оптимизация, алгоритм муравьиной колонии, NP-трудная задача, управление ресурсами, смешанные графы, календарное планирование

* *Статья получена 20 марта 2026 г.*

ВВЕДЕНИЕ

Сегодня разработки программных и инженерных систем всё чаще планируются с помощью проектного подхода. Суть такого планирования состоит в том, что множество одновременно выполняющихся проектов состоит из каких-либо задач, которые должны выполнить ограниченное число исполнителей. При этом исполнитель является ресурсом, без которого задача не может быть начата. Такая мультипроектная среда характеризуется ограниченностью ресурсов, конкуренцией между проектами и необходимостью эффективного планирования сроков исполнения проектов [1, 2].

Ресурс в задачах планирования и управления проектами понимается как любой ограниченный объект, средство или субъект, необходимый для выполнения задачи. Использование ресурса связано с ограничениями по его количеству, доступности или времени применения. Задача может быть начата только при наличии необходимых ресурсов, поэтому в мультипроектной среде несколько задач могут одновременно конкурировать за одни и те же ресурсы. Это приводит к необходимости их распределения во времени и между различными проектами.

Можно выделить разделяемые и неразделимые ресурсы [3]. Разделяемые ресурсы могут использоваться несколькими задачами одновременно, при этом их общий объем распределяется между задачами. Каждая задача получает лишь часть ресурса, а суммарное использование не должно превышать его доступный объем. К таким ресурсам относятся, например, вычислительные мощности, финансовые средства или пропускная способность каналов связи. Разделяемые ресурсы могут быть как складируемыми, так и нескладируемыми. Складируемые ресурсы обладают свойством накопления, т. е. неиспользованный в данный момент ресурс может быть сохранен и использован позже. Примером таких ресурсов могут служить материалы, комплектующие или финансовые средства. Нескладируемые ресурсы, напротив, не могут накапливаться: если они не используются в определенный момент времени, возможность их использования теряется. К таким ресурсам относится, например, рабочее время специалистов или вычислительное время оборудования.

Неразделимые ресурсы характеризуются тем, что в один момент времени они могут быть использованы только одной задачей. Если такой ресурс занят выполнением некоторой задачи, остальные задачи вынуждены ожидать его освобождения. В качестве примеров неразделимых ресурсов можно рассматривать конкретных специалистов, производственное оборудование или уникальные технические установки. Наличие неразделимых ресурсов в системе приводит к возникновению конфликтов доступа между задачами и существенно усложняет процесс планирования [4].

Исполнитель в нашей задаче является неразделимым ресурсом. Это означает, что только после окончания текущей операции исполнитель начнет следующую. Таким образом, получается, что помимо технологической последовательности есть также ресурсная.

Основной сложностью мультипроектного управления является распределение во времени ресурсов между задачами различных проектов с целью минимизации общего времени выполнения и повышения эффективности использования ресурсов. Эта задача относится к классу комбинаторных оптимизационных задач и является NP-трудной (Nondeterministic Polynomial-Time Hard – недетерминированно полиномиально трудная), т. е. задачей, для которой

не существует известного алгоритма с полиномиальной зависимостью времени решения от размерности задачи [5].

В последние годы для решения подобных задач активно применяются методы метаэвристической оптимизации. Одним из наиболее перспективных является муравьиный алгоритм (Ant Colony Optimization), основанный на моделировании поведения колоний муравьев при поиске кратчайших путей.

Муравьиный алгоритм напрямую относится к направлению роевого интеллекта и является одним из наиболее известных его представителей. Роевой интеллект представляет собой направление искусственного интеллекта, основанное на моделировании коллективного поведения децентрализованных систем, состоящих из большого числа относительно простых агентов. Такие системы не имеют централизованного управления, однако за счет взаимодействия между агентами способны демонстрировать сложное и организованное поведение [6].

Такой принцип коллективного поиска решения лежит в основе муравьиного алгоритма. В алгоритме множество искусственных агентов (муравьев) параллельно строят возможные решения задачи, используя информацию о накопленном феромоне и локальные эвристики. Взаимодействие агентов осуществляется через общую структуру данных, в которой хранится уровень феромона на элементах решения. Благодаря этому возникает механизм положительной обратной связи: решения, приводящие к лучшим значениям целевой функции, усиливаются и становятся более вероятными для выбора в последующих итерациях.

Целью настоящей работы является усовершенствование формальной постановки задачи мультипроектного планирования с учетом ресурсных ограничений и разработка эффективного алгоритма ее решения в парадигме роевого интеллекта.

1. ОБЗОР СУЩЕСТВУЮЩИХ ПОДХОДОВ

В статье [7] рассматривается подход к решению задачи календарного планирования с ограниченными ресурсами (Resource-Constrained Project Scheduling Problem – RCPSP), в которой необходимо определить моменты начала выполнения работ таким образом, чтобы были соблюдены технологические зависимости между работами и ограничения на использование ресурсов, а общий срок завершения проекта был минимальным. Для решения этой задачи авторы предлагают использовать генетический алгоритм, адаптированный с учетом особенностей ресурсных ограничений.

Предложенный подход основан на представлении возможного расписания работ в виде хромосомы, которая кодирует порядок выполнения операций проекта. Такой порядок не является непосредственно готовым расписанием, а используется как приоритетный список, на основе которого строится допустимое расписание с учетом всех технологических и ресурсных ограничений. Для преобразования хромосомы в расписание применяется процедура построения графика, которая последовательно назначает моменты начала работ, учитывая их длительность, зависимости предшествования и доступность ресурсов. В результате из каждого генетического решения формируется корректное расписание проекта, для которого вычисляется значение целевой функции – длина расписания или время завершения всех работ.

Основным недостатком такого подхода является его вычислительная сложность, обусловленная необходимостью построения допустимого расписания

для каждой хромосомы в популяции на каждой итерации. Процедура декодирования, учитывающая ресурсные ограничения и технологические зависимости, требует значительных вычислительных ресурсов, что критично при увеличении размерности задачи. Кроме того, эффективность генетического алгоритма сильно зависит от выбора параметров эволюции (размер популяции, вероятность мутации и скрещивания), а также от способа кодирования решений, что требует предварительной настройки и может приводить к преждевременной сходимости к субоптимальным решениям.

В статье [8] представлен подход к решению NP-трудной задачи календарного планирования типа job shop, основанный на адаптивной модификации метода роя частиц. Суть подхода заключается в создании двухуровневой иерархической системы оптимизации, где один рой частиц решает исходную задачу, а другой рой динамически настраивает его параметры для повышения эффективности поиска.

Для применения метода роя частиц, который изначально разработан для непрерывных пространств, к дискретной задаче составления расписания авторы используют механизм декодирования. Координаты каждой частицы нижнего уровня интерпретируются как приоритеты операций, на основе которых строится допустимое расписание и вычисляется его качество.

Верхний уровень системы реализует адаптивную настройку параметров основного алгоритма. Рой частиц верхнего уровня подбирает такие значения коэффициентов, как инерция и ускорение, которые максимизируют эффективность поиска роя нижнего уровня. Оценкой для частиц верхнего уровня служит качество решений, найденных основным алгоритмом с соответствующим набором параметров.

Недостатком предложенной двухуровневой иерархической системы является ее высокая ресурсоемкость, поскольку она требует параллельной работы двух роев частиц, что удваивает вычислительные затраты по сравнению с классическим методом. Эффективность подхода также критически зависит от корректности механизма декодирования непрерывных координат частиц в дискретные приоритеты операций, так как неудачное отображение может привести к потере информации и снижению качества решений. Кроме того, несмотря на адаптивную настройку, алгоритм сохраняет чувствительность к начальным значениям параметров верхнего уровня, которые задаются в начале работы.

В статье [9] рассматривается метод, основанный на итеративном улучшении исходного допустимого расписания. На каждом шаге алгоритм генерирует небольшое возмущение текущего решения, например, меняя порядок задач или переназначая ресурсы, и оценивает полученный вариант. Если новое расписание оказывается лучше с точки зрения времени выполнения, оно принимается. В случае ухудшения решение также может быть принято, но с вероятностью, зависящей от значения ухудшения и текущего значения управляющего параметра, называемого температурой. На начальных этапах температура высокая, что позволяет алгоритму активно исследовать пространство решений, не застревая в локальных оптимумах. Постепенно температура снижается, и алгоритм переходит в режим локальной оптимизации, отдавая предпочтение только улучшающим изменениям. Такая динамика обеспечивает баланс между глобальным поиском и локальной сходимостью, и в результате после большого числа итераций удается получить расписание, близкое к оптимальному и превосходящее по качеству простые эвристики.

К достоинствам метода можно отнести способность преодолевать локальные минимумы за счет вероятностного принятия ухудшающих решений на

ранних стадиях, что выгодно отличает его от жадных алгоритмов. Кроме того, алгоритм хорошо масштабируется и допускает гибкую настройку вычислительных ресурсов через ограничение числа итераций и выбор закона изменения температуры.

Вместе с тем подход не гарантирует достижения глобального оптимума и чувствителен к выбору параметров. Особенно важна корректная настройка механизма генерации соседних решений: слишком малые изменения замедляют сходимость, а чрезмерно крупные приводят к неэффективному перебору и росту числа неудачных попыток. Это требует адаптации алгоритма под конкретную задачу, что может усложнить его практическое применение.

2. ПОСТАНОВКА ЗАДАЧИ

Имеется P проектов, содержащих в сумме I задач (операций), также имеется ограниченное число исполнителей E. Каждая задача относится к одному проекту и выполняется одним исполнителем. Для описания операций можно воспользоваться тройной индексацией (p, e, q), где p – номер проекта, e – номер исполнителя, q – кратность использования, т. е. операция (p, e, q) выполняется e-м исполнителем в p-м проекте в q-й раз.

Однако для упрощения записи перейдем от тройной индексации к одинарной: $(p,\ e,\ q) \Leftrightarrow (i)$. Для всех задач операции пронумерованы $i = \overline{1, I}$. Известно t_i^n – время выполнения в нормальном режиме, также известна технологическая последовательность выполнения задач в проектах. Задачи, которые выполняются одним исполнителем, не могут происходить одновременно.

Необходимо найти решение задачи с критерием минимального времени окончания всех проектов.

В таком случае формальная постановка задачи [10] может быть представлена следующим образом. Целевая функция имеет вид

$$T_k \to \min \tag{1}$$

при следующих ограничениях:

$$T_j + t_j^n \le T_k \quad \text{при} \quad j = \overline{1, I}, \tag{2}$$

$$T_i + t_i^n \le T_j, \text{ если операция } i \text{ предшествует } j,\ i, j = \overline{1, I}, \tag{3}$$

$$T_j \ge 0 \quad \text{при} \quad j = \overline{1, I}, \tag{4}$$

$$\begin{cases} T_j - T_i - Bw_{i,j} \le -t_i^n, \\ T_i - T_j + Bw_{i,j} \le B - t_j^n, \end{cases} \tag{5}$$

если операции i и j не могут выполняться одновременно, $w_{i,j}$ – переменная порядка выполнения пары операций (i, j):

$$w_{i,j} = \begin{cases} 1, \text{ если операция } j \text{ выполняется после } i, \\ 0 \text{ в противном случае,} \end{cases} \tag{6}$$

где T_j – время начала операции j; T_k – время завершения последнего из проектов; $B > 0$ – достаточно большое положительное число, используемое для взаимоисключающего выполнения операций.

Ограничение (3) для всех технологических зависимостей гарантирует, что если операция j не может начаться до завершения операции i согласно технологическому процессу проекта, то момент начала T_j будет не меньше, чем момент начала T_i, увеличенный на время выполнения t_i^n операции i. Ограничение (4) задает естественное условие неотрицательности моментов начала всех операций. Ограничение (5) связывает временные переменные с переменными порядка: если операция i предшествует операции j на одном ресурсе $(w_{i,j}=1)$, то ее завершение должно наступать раньше начала j; если же $w_{i,j}=0$, то большая константа B делает это неравенство автоматически выполненным, не накладывая ограничений.

Эту задачу можно показать посредством смешанного графа для небольшого примера мультипроектной разработки (рис. 1).

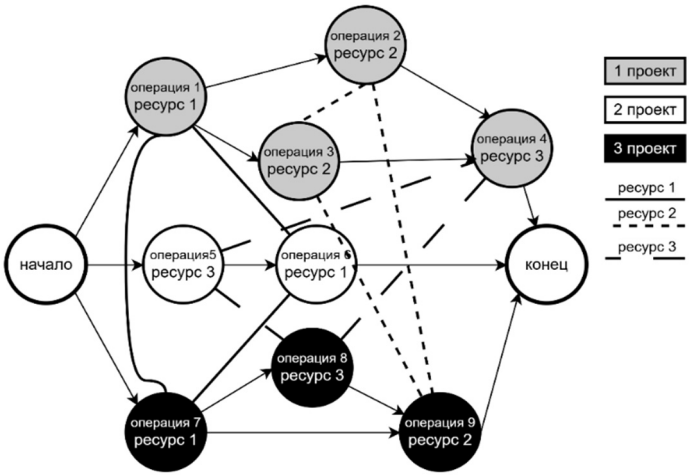


Рис. 1. Задача мультипроектных разработок в виде сетевого графа

Fig. 1. A multi-project development task in the form of a network graph

На сетевом графе вершины отображают операции, а технологические последовательности по проектам представлены дугами (связи со стрелками). Операции с одним ресурсом (исполнителем) соединены неориентированными ребрами, так как определенного порядка между ними в начале анализа нет. На рисунке операции с разными ресурсами соединены разными ребрами: ресурс 1 – жирной линией, ресурс 2 – мелким пунктиром, ресурс 3 – крупным пунктиром.

Используемая модель относится к классу смешанных, или дизъюнктивных, графов. Смешанный граф представляет собой граф, в котором одновременно присутствуют ориентированные дуги и неориентированные ребра. Ориентированные дуги задают жестко фиксированные технологические зависимости между операциями, определяющие допустимую последовательность их выполнения. Такие дуги отражают порядок выполнения работ внутри каждого проекта и формируют технологическую структуру задач. Неориентированные ребра, в свою очередь, отражают ресурсные ограничения и соединяют операции, которые используют один и тот же ресурс. Между такими операциями

на этапе построения модели порядок выполнения заранее не задан, поэтому соответствующие связи являются дизъюнктивными и должны быть ориентированы в процессе построения расписания.

В задачах теории расписаний смешанные графы позволяют одновременно учитывать технологические зависимости между операциями и ограничения по ресурсам. Формирование допустимого расписания в таком случае сводится к выбору направления для дизъюнктивных ребер, т. е. к определению порядка выполнения операций, конкурирующих за один и тот же ресурс. После ориентации этих ребер граф становится ориентированным ациклическим графом, по которому можно определить моменты начала выполнения операций и длительность всего расписания. Таким образом, использование смешанного графа позволяет наглядно представить структуру мультипроектной задачи и формализовать процесс поиска оптимального или близкого к оптимальному календарного плана выполнения работ.

3. АЛГОРИТМ МУРАВЬИНОЙ КОЛОНИИ

В природных условиях муравьи способны находить кратчайшие пути между муравейником и источником пищи, несмотря на ограниченные возможности отдельной особи. Обмен информацией между муравьями осуществляется косвенно – посредством выделения специального химического вещества, называемого феромоном. Передвигаясь по маршруту, муравьи оставляют феромон, который служит сигналом для других особей. Чем больше муравьев проходит по определенному пути, тем выше концентрация феромона и тем более привлекательным становится данный маршрут [11].

Феромон со временем испаряется, поэтому менее эффективные маршруты постепенно теряют привлекательность. В результате совместного действия процессов накопления и испарения феромона колония постепенно концентрирует движение на наиболее коротких и выгодных путях [12].

Популяция состоит из l искусственных муравьев, каждый из которых строит последовательность операций на каждом ресурсе заданное количество итераций N. На рис. 2 представлен пример решения конфликта на ресурсе 1. Ребра становятся дугами.

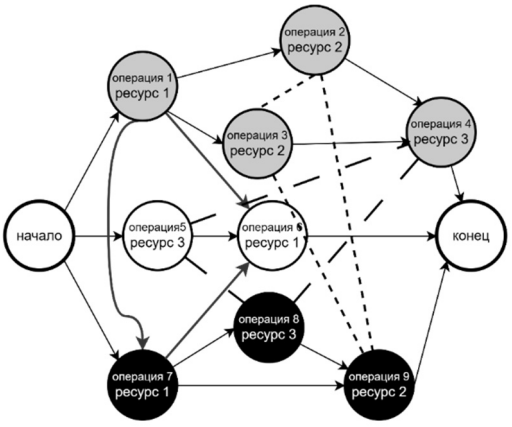


Рис. 2. Решение задачи на ресурсе 1

Fig. 2. Solution to the problem on resource 1

Можем заметить, что дуга «операция 1 – операция 6» является фиктивной, операция 6 всё равно будет осуществляться после операции 7, так как та идет после операции 1. В таком случае можно сделать вывод, что муравью достаточно построить прямую последовательность операций для каждого вида ресурса [1, 6, 7], подобно задаче коммивояжера.

Следовательно, у каждого муравья для каждого ресурса будет список с еще не посещенными вершинами. Выбор следующей вершины для посещения осуществляется по вероятностному правилу перехода:

$$p_{i,j} = \frac{\tau_{i,j}^{\alpha} d_{i,j}^{\beta}}{\sum\limits_{j \in V} \tau_{i,j}^{\alpha} d_{i,j}^{\beta}},$$

здесь $d_{i,j}$ – эвристическое расстояние между операциями i и j:

$$d_{i,j} = \frac{1}{T_j + 1},$$

где $p_{i,j}$ – вероятность перехода из вершины i в вершину j; V – вершины, доступные для посещения из вершины i (еще не посещенные операции, использующие тот же ресурс); $\tau_{i,j}$ – уровень феромона на ребре i, j; α – параметр, определяющий влияние феромона; β – параметр, определяющий влияние эвристической информации.

В рамках алгоритма ACO каждый искусственный муравей формирует допустимое расписание выполнения операций, последовательно выбирая операции из множества доступных к выполнению. Доступной считается операция, для которой выполнены все технологические предшествования и которая может быть назначена на соответствующий ресурс без нарушения ресурсных ограничений. При построении решения муравей выбирает следующую операцию i из множества допустимых операций A с вероятностью, зависящей от значения феромона $\tau_{i,j}$, связанного с переходом между операциями, и эвристической информации $d_{i,j}$, отражающей предпочтительность выбора операции, в нашем случае на основе времени начала операции. Можно эвристическую информацию выбирать на основе продолжительности операции, однако в случае, когда мы пересчитываем время после построения последовательности операций для каждого ресурса, предпочтительнее опираться на время начала, чтобы не было больших простоев. Вероятность выбора определяется стандартным соотношением алгоритма муравьиной колонии, в котором параметры α и β задают относительное влияние феромонной информации и эвристической оценки.

После того как муравей сформировал полную последовательность выполнения операций, на ее основе строится расписание с учетом технологических зависимостей и ограничений по ресурсам. Для каждой операции определяется момент начала выполнения как максимальное значение между моментом завершения всех ее технологических и ресурсных предшественников, затем расчет времени окончания всех проектов (самого позднего проекта) T_k. Если T_k меньше лучшего решения (или лучшего решения еще нет), то данная последовательность становится лучшим решением.

По завершении итерации алгоритма происходит обновление феромонов на дугах графа. Решения, приводящие к меньшей длительности общего расписания, усиливают соответствующие феромонные следы, что повышает вероятность выбора аналогичных последовательностей операций в последующих итерациях. Таким образом, алгоритм постепенно концентрирует поиск в области наиболее перспективных вариантов упорядочивания операций, что позволяет находить эффективные расписания выполнения мультипроектных разработок с учетом технологических и ресурсных ограничений. Сначала происходит процесс испарения феромонов:

$$\tau_{i,j} = \tau_{i,j}(1-\rho),$$

где ρ – коэффициент испарения феромона (от нуля до единицы включительно).

Происходит локальное добавление феромона на ребрах, входящих в найденные решения:

$$\tau_{i,j} = \tau_{i,j} + \Delta\tau_{i,j},$$

$$\Delta\tau_{i,j} = \begin{cases} \dfrac{Q}{T_k}, \text{ если } w_{i,j} = 1, \\ 0 \text{ в противном случае,} \end{cases}$$

где Q – константа, определяющая количество добавляемого феромона.

Значение феромона должно удовлетворять условию

$$\tau_{\min} \le \tau_{i,j} \le \tau_{\max},$$

где $\tau_{\min}$ – минимальный уровень феромона; $\tau_{\max}$ – максимальный уровень феромона.

Временную сложность можно рассчитать по формуле

$$TC = Nl\left(\sum_{e=1}^{E} n^2 + \varphi\right), \tag{7}$$

где TC – временная сложность (Time Complexity), N – количество итераций; l – количество муравьев; n – количество операций, использующих ресурс e; φ – время расчета временных характеристик.

Следовательно, временная сложность определяется совокупностью числа итераций и количества задействованных муравьев, а также затратами на обработку ресурсов в ходе вычислений. Полученное выражение позволяет выявить факторы, влияющие на производительность алгоритма и проанализировать характер роста вычислительной сложности.

4. ЭКСПЕРИМЕНТАЛЬНЫЕ ИССЛЕДОВАНИЯ ПРИМЕНЕНИЯ МУРАВЬИНОГО АЛГОРИТМА В ОПТИМИЗАЦИИ МУЛЬТИПРОЕКТНОГО ПЛАНИРОВАНИЯ

Для начала изучим влияние параметров на работу алгоритма. Для этого была взята задача с 36 операциями, состоящая из шести проектов и шести ресурсов с оптимальным решением (временем окончания), равным 132. Коли-

чество муравьев на всех запусках программы будет фиксированным (l = 36), коэффициент испарения ρ = 0,01, а коэффициенты α и β будем менять, чтобы понять, при каких их значениях за сколько итераций достигается оптимум. Результаты представлены в табл. 1.

Таблица 1

Table 1

Количество итераций, необходимое для достижения оптимума

The number of iterations required to reach the optimum

β \ α	α = 0	α = 1	α = 2	α = 3	α = 4	α = 5	α = 6	α = 7	α = 8
β = 0	>1000	>1000	>1000	>1000	>1000	>1000	>1000	>1000	>1000
β = 1	>1000	98	163	102	82	72	>1000	>1000	>1000
β = 2	10	1	1	21	16	11	2	11	6
β = 3	3	5	2	6	2	2	9	1	4
β = 4	1	1	1	1	1	1	1	1	1
β = 5	1	1	1	1	1	1	1	1	1
β = 6	1	1	1	1	1	1	1	1	1
β = 7	1	1	1	1	1	1	1	1	1
β = 8	1	1	1	1	1	1	1	1	1

Проведенный эксперимент позволил оценить влияние параметров алгоритма муравьиной колонии на скорость нахождения оптимального решения. Полученные результаты показывают, что при $\beta = 0$ алгоритм не способен найти оптимальное решение даже за 1000 итераций независимо от значения параметра α. Это объясняется тем, что при $\beta = 0$ муравьи не учитывают эвристическую информацию в выборе следующего перехода и ориентируются только на уровень феромона, который на начальных этапах практически одинаков для всех путей. В результате поиск становится близким к случайному и эффективность алгоритма резко снижается.

При небольших значениях параметра β (например, $\beta = 1$) алгоритм начинает учитывать эвристическую информацию, однако скорость сходимости остается относительно низкой. В этом случае оптимальное решение достигается за десятки или сотни итераций, а при некоторых значениях параметра α оптимум не достигается даже за 1000 итераций. Это указывает на то, что влияние эвристической составляющей в данном случае недостаточно для эффективного направленного поиска.

С увеличением значения параметра β наблюдается значительное улучшение результатов. Уже при $\beta = 2$ оптимальное решение достигается за небольшое количество итераций, обычно от одной до нескольких десятков. Это свидетельствует о том, что усиление влияния эвристической информации позволяет алгоритму быстрее концентрировать поиск в перспективных областях пространства решений.

При $\beta \geq 4$ алгоритм демонстрирует наилучшие результаты: оптимальное решение находится уже на первой итерации практически при всех значениях

параметра α. Это говорит о том, что при достаточно высоком влиянии эвристической информации поиск становится более направленным и алгоритм быстро находит оптимальный маршрут.

Таким образом, результаты эксперимента показывают, что параметр β оказывает существенно большее влияние на скорость сходимости алгоритма, чем параметр α. Для рассматриваемой задачи наилучшие результаты достигаются при значениях β не менее 4, при которых алгоритм стабильно и быстро находит оптимальное решение независимо от выбранного значения параметра α. Полученные результаты подтверждают необходимость корректной настройки параметров алгоритма для обеспечения его высокой эффективности при решении задач календарного планирования.

Далее алгоритм тестировался на трех задачах по три раза (так как алгоритм содержит случайную компоненту) и сравнивался с результатами, полученными в IBM ILOG CPLEX Optimization Studio (программный комплекс для решения масштабных задач оптимизации) (табл. 2).

Название задач представлено в виде *I* – *P* – *E* (количество исполнителей), где *I* – количество операций, *P* – количество проектов, *E* – количество исполнителей. Точнее говоря, в задаче 25–5–5 всего 25 операций, 5 проектов и 5 ресурсов; в задаче 100–10–10 операций 100, проектов 10, ресурсов 10; задача 121–11–11 включает 121 операцию, 11 проектов и 11 ресурсов.

Проведенные эксперименты позволили оценить эффективность применения муравьиного алгоритма для решения задачи мультипроектного планирования и сравнить полученные результаты с решениями, найденными с использованием системы оптимизации IBM ILOG CPLEX Optimization Studio.

Таблица 2

Table 2

Результаты тестирования алгоритма

Algorithm testing results

Задача	CPLEX		Муравьиный алгоритм		
	T_k	Время решения	T_k	Отклонение от оптимума	Время решения
25–5–5	97	00:00:04	97	0 %	00:00:00
			97	0 %	00:00:00
			97	0 %	00:00:00
100–10–10	226	00:00:06	226	0 %	00:00:17
			234	3,5 %	00:00:10
			229	1,3 %	00:00:10
121–11–11	322	> 05:00:00	322	0 %	00:00:07
			322	0 %	00:00:02
			322	0 %	00:00:02

В качестве иллюстрации получаемых решений рассмотрим пример расписания, построенного алгоритмом для задачи небольшой размерности. Для задачи 25–5–5 алгоритм сформировал следующую последовательность выполнения операций (рис. 3).

```
=== РЕЗУЛЬТАТЫ ===
Время окончания всех проектов (Tk) = 97
Операция № 1: время начала (Ti) = 0, продолжительность = 12
Операция № 2: время начала (Ti) = 30, продолжительность = 9
Операция № 3: время начала (Ti) = 39, продолжительность = 8
Операция № 4: время начала (Ti) = 49, продолжительность = 22
Операция № 5: время начала (Ti) = 73, продолжительность = 7
Операция № 6: время начала (Ti) = 19, продолжительность = 11
Операция № 7: время начала (Ti) = 30, продолжительность = 12
Операция № 8: время начала (Ti) = 42, продолжительность = 21
Операция № 9: время начала (Ti) = 63, продолжительность = 24
Операция № 10: время начала (Ti) = 90, продолжительность = 5
Операция № 11: время начала (Ti) = 0, продолжительность = 19
Операция № 12: время начала (Ti) = 19, продолжительность = 23
Операция № 13: время начала (Ti) = 47, продолжительность = 13
Операция № 14: время начала (Ti) = 60, продолжительность = 5
Операция № 15: время начала (Ti) = 80, продолжительность = 10
Операция № 16: время начала (Ti) = 0, продолжительность = 15
Операция № 17: время начала (Ti) = 19, продолжительность = 8
Операция № 18: время начала (Ti) = 42, продолжительность = 7
Операция № 19: время начала (Ti) = 63, продолжительность = 17
Операция № 20: время начала (Ti) = 80, продолжительность = 8
Операция № 21: время начала (Ti) = 0, продолжительность = 19
Операция № 22: время начала (Ti) = 39, продолжительность = 22
Операция № 23: время начала (Ti) = 65, продолжительность = 8
Операция № 24: время начала (Ti) = 73, продолжительность = 17
Операция № 25: время начала (Ti) = 90, продолжительность = 7
Ресурс № 0; последовательность операций = 1, 8, 19, 15, 25
Ресурс № 4; последовательность операций = 11, 6, 2, 22, 20
Ресурс № 2; последовательность операций = 21, 17, 3, 13, 9
Ресурс № 1; последовательность операций = 12, 18, 4, 24, 10
Ресурс № 3; последовательность операций = 16, 7, 14, 23, 5
Время выполнения программы: 0,421 сек.
```

Рис. 3. Решение задачи 25 – 5 – 5

Fig. 3. Solution to problem 25 – 5 – 5

Решение этой же задачи в CPLEX:

// solution (optimal) with objective 97
// Quality Incumbent solution:
// MILP objective 9,7000000000e+01
// MILP solution norm |x| (Total, Max) 1,26900e+03 9,70000e+01

x = [0 30 39 51 80 19 30 42 66 90 0 19 47 75 80 0 31 42 63 80 0 39 61 73 90 97];
w = [0 0 0 0 0 0 0 0 1 0 0 1 1 0 1 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 1 0 1 1 0 0 1 0 0 0 0 1];

В обоих случаях значение целевой функции совпадает. Время завершения всех проектов составляет $T_k = 97$ (временных единиц). Это означает, что разработанный муравьиный алгоритм смог найти решение, совпадающее с оптимальным решением, полученным методом точной оптимизации в CPLEX. Следовательно, по критерию длительности расписания результаты алгоритма полностью соответствуют оптимальному значению.

При сравнении координат переменных, т. е. времени начала отдельных операций, можно заметить, что для части операций значения полностью совпадают. Например, операции 1, 2, 3, 6, 7, 8, 10, 11, 12, 13, 16, 18, 22, 24 и 25 имеют одинаковое время начала в обоих решениях. Для некоторых операций наблюдаются небольшие различия. Так, операция 4 в муравьином алгоритме начинается в момент времени 49, тогда как в CPLEX – в момент 51, операция 5 – 73 против 80, операция 9 – 63 против 66, операция 14 – 60 против 75,

операция 17 – 19 против 31, операция 19 – 49 против 63, операция 20 – 66 против 80, операция 23 – 65 против 61.

Такие различия объясняются тем, что задача допускает существование нескольких различных расписаний с одинаковым значением целевой функции. При этом суммарная длительность выполнения всех проектов остается неизменной. Муравьиный алгоритм формирует одно из допустимых расписаний, удовлетворяющих всем ограничениям по ресурсам и технологическим зависимостям, тогда как CPLEX находит одно из оптимальных решений задачи линейного целочисленного программирования.

Анализ результатов показывает, что для задачи небольшого размера (25–5–5) муравьиный алгоритм во всех запусках смог найти оптимальное решение, совпадающее с результатом CPLEX. При этом время работы алгоритма оказалось крайне малым и практически мгновенным, что свидетельствует о высокой эффективности метода для задач небольшой размерности.

Для более наглядной оценки посчитаем количество ограничений, непрерывных и булевых переменных. Так как операций 25, то непрерывных переменных, обозначающих время начала операций и окончания операций, в два раза больше:

$$S = 2I,$$

где S – количество непрерывных переменных.

Количество булевых переменных (6), которые необходимо найти, можно рассчитать по формуле

$$W = \sum_{e=1}^{E} C_n^2 = \sum_{e=1}^{E} \frac{n(n-1)}{2},$$

где W – количество переменных $w_{i,j}$ порядка выполнения для всех операций i, j, которые не могут выполняться одновременно; n – количество операций, использующих ресурс e.

Количество ограничений определяется по формуле

$$C = C_{\text{техн}} + C_{\text{неотр}} + C_{\text{рес}} + C_{\text{цел}}, \tag{8}$$

$$C_{\text{техн}} = I - P,$$

$$C_{\text{неотр}} = I,$$

$$C_{\text{рес}} = 2W,$$

$$C_{\text{цел}} = I,$$

где C – общее количество ограничений для задачи; $C_{\text{техн}}$ – количество технологических ограничений (3); $C_{\text{неотр}}$ – количество ограничений неотрицательности для моментов начала операций (5); $C_{\text{рес}}$ – количество ограничений, связанных с ресурсными ограничениями (6); $C_{\text{цел}}$ – количество ограничений, связанных с целевой функцией (2).

Подставив эти выражения в формулы (8), получим общую формулу количества ограничений в упрощенном виде:

$$C = 3I - \mathrm{P} + 2W.$$

Следовательно, для задачи 25–5–5 $S = 50,\ W = 10,\ C = 90.$

Для задачи большого размера (100–10–10) результаты оказались более вариативными. В одном из запусков муравьиный алгоритм также достиг оптимального решения, совпадающего с результатом CPLEX. В других запусках были получены решения с небольшим отклонением от оптимума – 3,5 % и 1,3 %. При этом время работы алгоритма составило от 10 до 17 секунд, что превышает время решения CPLEX, однако отклонение от оптимального значения остаётся относительно небольшим. Это подтверждает, что муравьиный алгоритм способен находить решения хорошего качества даже для более сложных задач, хотя из-за стохастической природы метода результаты отдельных запусков могут отличаться. Для данной задачи $S = 200,\ W = 450,\ C = 1190,$ что существенно больше, чем в задаче 25–5–5 и усложняет поиск оптимального решения.

Наиболее показательные результаты были получены для задачи еще большего размера (121–11–11). В данном случае точный метод, реализованный в CPLEX, не смог найти решение в течение более чем пяти часов вычислений, поэтому мы воспользовались CP, реализованный в модуле IBM ILOG CP Optimizer [13]. CP (Constraint Programming – программирование с ограничениями) представляет собой метод комбинаторной оптимизации, основанный на задании системы ограничений, применении специализированных процедур распространения ограничений и поиска по дереву решений. Такой подход широко применяется для решения задач календарного планирования и дискретной оптимизации. В результате использование CP позволило получить решение за 0,74 секунды с отклонением 0 % (оптимум). Муравьиный алгоритм во всех трех запусках также смог получить оптимальное значение целевой функции за несколько секунд. Это демонстрирует одно из ключевых преимуществ метаэвристических методов: способность эффективно находить хорошие решения для задач большой размерности, для которых точные методы требуют значительных вычислительных ресурсов или становятся практически неприменимыми. Для данной задачи $S = 242,\ W = 679,\ C = 1710.$ При таком количестве переменных и ограничений точные методы становятся практически неприменимы из-за комбинаторной сложности [14].

Таким образом, результаты экспериментов показывают, что муравьиный алгоритм является эффективным инструментом для решения задач мультипроектного планирования. Для задач малой размерности он способен находить оптимальные решения за минимальное время, а для более сложных задач обеспечивает получение решений, близких к оптимальным, при существенно меньших вычислительных затратах по сравнению с точными методами оптимизации.

В рамках проведенных вычислительных экспериментов была оценена вычислительная трудоемкость алгоритма муравьиной колонии. В задаче 121–11–11 использовалось $N = 10$ итераций и $l = 11$ муравьев при $I = 121$ операций. На каждой итерации каждый муравей последовательно формирует допустимое решение задачи, выбирая операции из множества доступных. На каждом шаге

выполняется вычисление временных характеристик расписания и проверка ограничений по ресурсам, что определяет трудоемкость одного шага алгоритма. Расчет временных характеристик φ после построения последовательности требует определения моментов начала всех операций с учетом технологических и ресурсных предшествований, что выполняется за $O(I^2)$ операций, так как для каждой операции необходимо проверить всех ее предшественников [15]. Так, трудоемкость одного шага для задачи 121–11–11 при равномерно распределенных операциях по ресурсам равна

$$T = \sum_{e=1}^{11} 11^2 + 121^2 = 1131 + 14\,641 = 15\,772 .$$

С учетом того, что в эксперименте использовалось $N = 10$ итераций и $l = 11$ муравьев, общая трудоемкость алгоритма для задачи размерности $I = 121$ составит

$$T_{\text{общ}} = Nl\left(\sum_{e=1}^{11} 11^2 + 121^2\right) = 10 \cdot 11 \cdot 15\,772 = 1\,734\,920 \text{ операций.}$$

Для оценки зависимости трудоемкости от размерности задачи проанализируем структуру полученной формулы (7). Слагаемое $\sum_{e=1}^{E} n^2$ характеризует затраты на упорядочивание операций в рамках отдельных ресурсов и в худшем случае не превышает I^2. Слагаемое φ, равное I^2, соответствует расчету временных характеристик расписания. Таким образом, оба слагаемых имеют порядок $O(I^2)$, и, следовательно, общая вычислительная сложность алгоритма составляет

$$TC = O(I^2).$$

Это означает, что алгоритм муравьиной колонии обладает полиномиальной временной сложностью: при увеличении числа операций в k раз время работы алгоритма увеличивается не более чем в k^2 раз.

Можно сделать вывод, что алгоритм муравьиной колонии обладает полиномиальной вычислительной трудоемкостью. Это означает, что время его работы увеличивается пропорционально росту основных параметров задачи, таких как число операций, количество итераций и число муравьев, участвующих в построении решений. Благодаря этому алгоритм демонстрирует достаточно высокое быстродействие и может эффективно применяться для любых реальных задач мультипроектного планирования с большим числом операций и ресурсных ограничений.

ЗАКЛЮЧЕНИЕ

В ходе работы была исследована возможность применения алгоритма муравьиной колонии для оптимизации распределения ресурсов в мультипроектной среде. Выполнена математическая постановка задачи с использованием

смешанных графов, учитывающих технологические зависимости и ресурсные ограничения. Разработанный алгоритм адаптирован для построения допустимых расписаний путем вероятностного выбора последовательности операций. Экспериментально подтверждена высокая эффективность подхода: на малых задачах достигается точное совпадение с оптимальными решениями CPLEX, на средних и больших – близкие к оптимуму результаты за приемлемое время, тогда как точные методы (и программные средства на их основе) становятся неприменимы. Полученные результаты позволяют рекомендовать алгоритм и разработанную его программную реализацию для практического использования в системах мультипроектного управления. Разработанный подход может быть применен в различных областях, где требуется планирование множества взаимосвязанных задач при ограниченных ресурсах: управление разработкой программного обеспечения, планирование производственных процессов, управление инженерными и строительными проектами, а также задачи распределения вычислительных ресурсов в информационных системах. Использование муравьиного алгоритма особенно целесообразно при большой размерности задачи, когда применение точных методов становится вычислительно затруднительным.

СПИСОК ЛИТЕРАТУРЫ

1. *Лазарев А.А., Гафаров Е.Р.* Теория расписаний. Задачи и алгоритмы. – М.: МГУ, 2011.

2. *Manne A.S.* On the job-shop scheduling problem // Operations Research. – 1960. – Vol. 8 (2). – P. 219–223.

3. *Олейникова С.А.* Математическая модель и оптимизационная задача составления расписания для мультипроектной системы с временными и ресурсными ограничениями и критерием равномерной загрузки // Вестник ВГТУ. – 2013. – № 6-3. – С. 58–61.

4. *Матвеев А.И.* Алгоритм оптимизации планирования ресурсов (на примере метода отжига) // Перспективные информационные технологии (ПИТ 2018). – Самара, 2018. – С. 1046–1050.

5. *Xiong H.* A survey of job shop scheduling problem: The types and models // Computers & Operations Research. – 2022. – Vol. 142. – P. 105731.

6. *Курейчик В.М., Кажаров А.А.* Использование роевого интеллекта в решении NP-трудных задач // Известия ЮФУ. Технические науки. – 2011. – № 7. – С. 30–36.

7. *Гончаров Е.Н., Леонов В.В.* Генетический алгоритм для задачи календарного планирования с ограниченными ресурсами // Автоматика и телемеханика. – 2017. – № 6. – С. 173–189.

8. *Блок И.Н., Секаев В.Г.* Разработка, исследование и адаптация метода роя частиц в задачах дискретной оптимизации теории расписаний // Глобальный научный потенциал. – 2015. – № 5. – С. 70–75.

9. *Chakraborty S., Bhowmik S.* Job shop scheduling using simulated annealing // First International Conference on Computation and Communication Advancement. – 2013. – Vol. 1 (1). – P. 69–73.

10. *Mezentsev Y.A., Chubko N.Y.* On one bicriterion discrete optimization problem and a hybrid ant colony algorithm for its approximate solution // Lecture Notes in Computer Science. – 2021. – Vol. 12689. – P. 289–300.

11. *Саймон Д.* Алгоритмы эволюционной оптимизации. – М.: ДМК Пресс, 2020. – 1002 с.

12. *Dorigo M., Stützle T.* Ant colony optimization: overview and recent advances // Handbook of Metaheuristics. – Springer, 2018. – P. 311–351.

13. IBM. Constraint programming in CP Optimizer. – URL: https://www.ibm.com/products/ilog-cplex-optimization-studio/cplex-cp-optimizer (accessed: 16.03.2026).

14. *Клеванский Н.Н., Красников А.А.* Формирование календарных графиков мультипроектного планирования // Образовательные ресурсы и технологии. – 2015. – № 3. – С. 11–33.

15. *Son P.V.H., Khoi L.N.Q., Loc L.X.* Multi-project scheduling optimization with artificial intelligence: a novel metaheuristic framework // Cluster Computing. – 2026. – Vol. 29. – P. 63.

Мезенцев Юрий Анатольевич, доктор технических наук, профессор кафедры автоматизированных систем управления Новосибирского государственного технического университета. Основные направления научных исследований – автоматика, вычислительная техника, компьютерные науки, математика, кибернетика. Имеет более ста печатных работ. E-mail: mesyan@yandex.ru

Борисова Алина Алексеевна, бакалавр кафедры автоматизированных систем управления Новосибирского государственного технического университета. Основное направление научных исследований – компьютерные науки. Имеет две печатные работы. E-mail: alina_10112004@mail.ru

Mezentsev Yurii A., PhD (Eng.), professor at the department of automated control systems, Novosibirsk State Technical University. The main areas of his scientific research are automation, computer engineering, computer science, mathematics, and cybernetics. He has more than a hundred printed works. E-mail: mesyan@yandex.ru

Borisova Alina A., bachelor at the department of automated control systems, Novosibirsk State Technical University. The main area of her scientific research is computer science. She has 2 printed works. E-mail: alina_10112004@mail.ru

DOI: 10.17212/2782-2001-2026-3-97-114

Application of the ant colony algorithm in optimizing the management of multi-project development[*]

Y. A. MEZENTSEV[a], *A. A. BORISOVA*[b]

Novosibirsk State Technical University, 20 Karl Marx Prospekt, Novosibirsk, 630073, Russian Federation

[a] *mesyan@yandex.ru* [b] *alina_10112004@mail.ru*

Abstract

The article discusses the application of the Ant Colony Optimization (ACO) algorithm to solve the problem of optimizing the management of multi-project developments with limited resources. A multi-project environment is characterized by competition between tasks for resources, technological and resource dependencies, which makes the task combinatorial and NP-hard.

The paper describes an algorithm for an ant colony which includes the formation of acceptable schedules in parallel with several artificial ants, a probabilistic choice of operations taking into account the pheromone level and heuristic information, as well as updating pheromone traces during iterations. Experiments have been conducted on problems of various dimensions, which have shown that with the correct parameter setting, the algorithm consistently finds optimal or close to optimal solutions. For small tasks, the solution completely coincides with the results of precise optimization in IBM ILOG CPLEX, for medium and large tasks, the method provides efficient search for high-quality schedules in significantly less time compared to precise methods.

The research results demonstrate the polynomial computational complexity of the ACO algorithm and its high efficiency in planning multi-project developments, which makes it a promising tool for use in practical tasks of managing project resources and deadlines.

Keywords: multi-project planning, combinatorial optimization, ant colony algorithm, NP-hard problem, resource management, mixed graphs, calendar planning

REFERENCES

1. Lazarev A.A., Gafarov E.R. *Teoriya raspisanii. Zadachi i algoritmy*. Moscow, Moscow State University Publ., 2011.

2. Mann A.S. On the task of planning working hours in the workshop. *Operation Research*, 1960, vol. 8 (2), pp. 219–223.

[*] *Received 20 March 2026.*

3. Oleynikova S.A. Matematicheskaya model' i optimizatsionnaya zadacha sostavleniya raspisaniya dlya mul'tiproektnoi sistemy s vremennymi i resursnymi ogranicheniyami i kriteriem ravnomernoi zagruzki [Mathematical model and optimization problem of scheduling for multi-project system with time and resource constraints and criteria of uniform downloads]. *Vestnik Voronezhskogo gosudarstvennogo tekhnicheskogo universiteta = Bulletin of Voronezh State Technical University*, 2013, no. 6-3, pp. 58–61.

4. Matveev A.I. [Resource planning optimization algorithm]. *Perspektivnye informatsionnye tekhnologii (PIT 2018)* [Advanced information technologies and scientific computing (PIT 2018)]. Samara, 2018, pp. 1046–1050. (In Rssian).

5. Xiong H. An overview of the problem of workplace planning in workshops: types and models. *Computers & Operations Research*, 2022, vol. 142, p. 105731.

6. Kureichik V.M., Kazharov A.A. Ispol'zovanie roevogo intellekta v reshenii NP-trudnykh zadach [The use of new intelligence in the analysis of NP-labor resources]. *Izvestiya Yuzhnogo federal'nogo universiteta. Tekhnicheskie nauki = Izvestiya Southem Federal University. Engineering sciences*, 2011, no. 7, pp. 30–36.

7. Goncharov E.N., Leonov V.V. Geneticheskii algoritm dlya zadachi kalendarnogo planirovaniya s ogranichennymi resursami [Genetic algorithm for the resource-constrained project scheduling problem]. *Avtomatika i telemekhanika = Automation and Remote Control*, 2017, no. 6, pp. 173–189. (In Russian).

8. Blok I.N., Sekaev V.G. Razrabotka, issledovanie i adaptatsiya metoda roya chastits v zadachakh diskretnoi optimizatsii teorii raspisanii [Development, research and adaptation of the method of the particles' swarm in discrete optimization problems of scheduling theory]. *Global'nyi nauchnyi potentsial = Global Scientific Potential*, 2015, no. 5, pp. 70–75.

9. Chakraborty S., Boumik S. Workflow planning using simulated annealing. *The First International Conference on the Development of Computer Technology and Communications*, 2013, vol. 1 (1), pp. 69–73.

10. Mezentsev Y.A., Chubko N.Y. On a two-criterion discrete optimization problem and a hybrid ant colony algorithm for its approximate solution. *Lecture Notes in Computer Science*, 2021, vol. 12689, pp. 289–300.

11. Simon D. *Algoritmy evolyutsionnoi optimizatsii* [Algorithms of evolutionary optimization]. Moscow, DMK Press, 2020. 1002 p. (In Russian).

12. Dorigo M., Stützle T. Optimization of the ant colony: an overview and recent achievements. *Handbook of Metaheuristics.* Springer, 2018, pp. 311–351.

13. IBM. *Constraint programming in CP Optimizer.* Available at: https://www.ibm.com/products/ilog-cplex-optimization-studio/cplex-cp-optimizer (accessed 16.03.2026).

14. Klevansky N.N., Krasnikov A.A. Formirovanie kalendarnykh grafikov mul'tiproektnogo planirovaniya [Multi-project scheduling problems techniques]. *Obrazovatel'nye resursy i tekhnologii = Educational Resources and Technologies*, 2015, no. 3, pp. 11–33.

15. Son P.V.H., Khoi L.N.Q., Loc L.X. Multi-project scheduling optimization with artificial intelligence: a novel metaheuristic framework. *Cluster Computing*, 2026, vol. 29, p. 63.

Для цитирования:

Мезенцев Ю.А., Борисова А.А. Применение алгоритма муравьиной колонии в оптимизации управления мультипроектными разработками // Системы анализа и обработки данных. – 2026. – № 3 (103). – С. 97–114. – DOI: 10.17212/2782-2001-2026-3-97-114.

For citation:

Mezentsev Y.A., Borisova A.A. Primenenie algoritma murav'inoi kolonii v optimizatsii upravleniya mul'tiproektnymi razrabotkami [Application of the ant colony algorithm in optimizing the management of multi-project development]. *Sistemy analiza i obrabotki dannykh = Analysis and Data Processing Systems*, 2026, no. 3 (103), pp. 97–114. DOI: 10.17212/2782-2001-2026-3-97-114.