PARALLEL IMPLEMENTATION OF THE ANT COLONY ALGORITHM WITH PARAMETERS UPDATE USING THE GENETIC ALGORITHM

I. Mikulik, E. Blagoveshchenskaya

Petersburg State Transport University,
190031, Saint Petersburg, Russia

DOI: 10.24412/2073-0667-2023-2-86-97
EDN: HBTPLC

The paper considers the possibility of using a joint implementation of a hybrid method using the ant colony optimization with genetic algorithm for solving traveling salesman problem. It is known that the ant colony optimization is sensitive to its parameters, so the search for the optimal parameters of the ant colony is suitable as a problem, the solution of which is related to the genetic algorithm. The one of the purposes of calculations parallelization is to reduce execution time, but not every algorithm has an effective parallel implementation. It is known that the genetic algorithm and the ant colony optimization are parallelized. The paper studies the possibility of constructing parallel computations for the hybrid method presented. The traveling salesman problem on which the research is conducted is an NP-complete problem and it is often used to test combinatorial optimization algorithms. It is shown that parallelization of the method used leads to an increase in the speed of the algorithm.

Key words: traveling salesman problem, optimization methods, ant colony optimization, genetic algorithm, parallel computing.

References

1. Zhang N. Moore's law is dead, long live moore's law! // arXiv preprint arXiv:2205.15011, 2022.

2. Pujol R., Jorba J., Tabani H., Kosmidis L., Mezzetti E., Abella J., Cazorla F. Vector extensions in cots processors to increase guaranteed performance in real-time systems, 2022. ACM Transactions on Embedded Computing Systems (TECS).

3. Zhou K. Feng X. A collaborative grouping aggregation query scheme on heterogeneous computing systems // in 2022 7th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA), 2022. P. 53–61.

4. Regassa D., Yeom H., Son Y. Harvesting the aggregate computing power of commodity computers for supercomputing applications // Applied Sciences, 2022. V. 12, N 10, P. 5113.

5. Ong B. W., Schroder J. B. Applications of time parallelization // Computing and Visualization in Science, 2020. V. 23, N 1. P. 1–15.

6. Blagoveshchenskaya E., Zuev D., Kuznecova I., Tihomirov S. Prilozheniya algoritmov pryamyh razlozhenij abelevyh grupp bez krucheniya k zadacham rasparallelivaniya vychislitel'nyh i konstruktivnyh processov // in Mezhdunarodnye Kolmogorovskie chteniya-XIV, posvyashchennye 100-letiyu professora Z. A. Skopeca, 2017. P. 38–40.

The research was supported by the Russian Science Foundation (project No. 22-21-00267).

7. Stutzle T. Parallelization strategies for ant colony optimization // in International Conference on Parallel Problem Solving from Nature. Springer, 1998. P. 722–731.

8. Hassan M., Hasan M., Hashem M. et al. An improved acs algorithm for the solutions of larger tsp problems. arXiv preprint arXiv:1304.3763, 2013.

9. Liang D., Zhan Z.-H., Zhang Y., Zhang J. An efficient ant colony system approach for new energy vehicle dispatch problem // IEEE Transactions on Intelligent Transportation Systems, 2019. V. 21. N 11. P. 4784–4797.

10. Zhou X., Ma H., Gu J., Chen H., Deng W. Parameter adaptation-based ant colony optimization with dynamic hybrid mechanism // Engineering Applications of Artificial Intelligence, 2022. V. 114. P. 105139.

11. Olivas F., Valdez F., Castillo O., Gonzalez C. I., Martinez G., Melin P. Ant colony optimization with dynamic parameter adaptation based on interval type-2 fuzzy logic systems // Applied Soft Computing, 2017. V. 53. P. 74–87.

12. Blagoveshchenskaya E. A., Mikulik I. I., Strëungmann L. H. Ant colony optimization with parameter update using a genetic algorithm for travelling salesman problem // Models and Methods for Researching Information Systems in Transport 2020 (MMRIST 2020), 2020. N 1. P. 20–25.

13. Baydogmus G. K. A parallelization based ant colony optimization for travelling salesman problem // in 2022 1st International Conference on Information System & Information Technology (ICISIT). IEEE, 2022. P. 166–169.

14. Liu G., Xu X., Wang F., Tang Y. Solving traveling salesman problems based on artificial cooperative search algorithm // Computational Intelligence and Neuroscience, 2022. V. 2022.

15. El-Khatib S. , Skobtsov Y. , Rodzin S. , Zakharov V. Comparison of modified object detected graph cut and hybrid ant colony optimization-k-means for mri images segmentation // in Computer Science On-line Conference. Springer, 2021. P. 701–708.

16. Brand M., Masuda M., Wehner N., Yu X.-H. Ant colony optimization algorithm for robot path planning // in 2010 international conference on computer design and applications, V. 3. IEEE, 2010. P. V3–436.

17. Whitley D. Next generation genetic algorithms: a user's guide and tutorial // in Handbook of metaheuristics. Springer, 2019. P. 245–274.

18. Katoch S., Chauhan S. S., Kumar V. A review on genetic algorithm: past, present, and future // Multimedia Tools and Applications, 2021. V. 80, N 5. P. 8091–8126.

19. Singh G., Gupta N. A study of crossover operators in genetic algorithms // in Frontiers in Nature-Inspired Industrial Optimization. Springer, 2022. P. 17–32.

20. Lambora A., Gupta K., Chopra K. Genetic algorithm-a literature review // in 2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon). IEEE, 2019. P. 380–384.

РАСПАРАЛЛЕЛИВАНИЕ ГИБРИДНОГО АЛГОРИТМА МУРАВЬИНОЙ КОЛОНИИ С ИЗМЕНЯЮЩИМИСЯ С ПОМОЩЬЮ ГЕНЕТИЧЕСКОГО АЛГОРИТМА ПАРАМЕТРАМИ

И. И. Микулик, Е. А. Благовещенская

Петербургский государственный университет путей сообщения
Императора Александра I,
190031, Санкт-Петербург, Россия

УДК 519.854.2
DOI: 10.24412/2073-0667-2023-2-86-97
EDN: HBTPLC

В работе рассмотрена возможность использования параллельной реализации гибридного метода оптимизации, использующего муравьиный и генетический алгоритмы, для решения задачи коммивояжера. Известно, что алгоритм муравьиной колонии чувствителен к изменению параметров, поэтому поиск подходящих параметров муравьиного алгоритма рассматривается в качестве задачи оптимизации, решение которой предоставляется генетическому алгоритму. Целью распараллеливания вычислений является сокращение временных затрат, но не все алгоритмы имеют эффективную параллельную реализацию. Известно, что генетический алгоритм и алгоритм муравьиной колонии распараллеливаются. В работе изучается возможность построения параллельных вычислений и для представленного гибридного метода. Задача коммивояжера, на которой проводится исследование, является NP-полной задачей и часто применяется для тестирования алгоритмов комбинаторной оптимизации. Показано, что распараллеливание используемого метода приводит к увеличению скорости выполнения работы алгоритма.

Ключевые слова: распараллеливание, задача коммивояжера, методы оптимизации, муравьиные алгоритмы, генетический алгоритм, параллельные вычисления.

Введение. Количество вычислительных узлов на микросхемах с каждым годом возрастает, в то время как замедляется рост частоты отдельно взятого процессора [1]. В связи с этим возникает потребность в выборе программных моделей, поддающихся эффективному с точки зрения использования вычислительных мощностей распараллеливанию.

Параллельные вычисления лежат в основе работы графических ускорителей и сопроцессоров [2–3]. Распараллеливание алгоритмов актуально и для работы с суперкомпьютером [4], который представляет собой вычислительное устройство, состоящее из большого количества узлов, соединенных магистральной шиной. Суперкомпьютеры находят свое приложение в решении задач численного моделирования в областях медицины, химии, физики [5]. Распараллеливание вычислительных процессов находит приложение и в фундаментальных областях науки, например в теории групп [6].

Исследование выполнено за счет гранта Российского научного фонда (проект № 22-21-00267).

Одним из классов алгоритмов, поддающихся эффективному распараллеливанию, является класс муравьиных алгоритмов [7]. Алгоритм муравьиной колонии является одним из эмпирических методов оптимизации и используется для решения комбинаторных задач. Он имеет практическое применение для задач прикладных областей: задач логистики [8], задач диспетчеризации [9], планирования маршрутов.

Общим недостатком методов класса муравьиных алгоритмов является чувствительность к параметрам, так как от них зависят качество решения и время, затраченное на его поиск. Оптимальные параметры могут быть специфичными для конкретной задачи и зависеть от требуемой точности решения, по этой причине существует множество модификаций методов муравьиной оптимизации, посвященных его динамической адаптации [10–11]. Данная работа посвящена возможности реализации одной из модификаций гибридного алгоритма муравьиной колонии, являющегося алгоритмом динамической адаптации параметров, впервые представленным в работе [12].

Целью настоящего исследования является ускорение работы рассматриваемого алгоритма посредством распараллеливания. Необходимо убедиться, что данный алгоритм имеет эффективную с точки зрения времени работы вычислительных узлов параллельную реализацию, также как и классические вариации алгоритма муравьиной колонии [13]. Для достижения поставленной цели в следующих разделах статьи будет представлено краткое описание муравьиного, генетического и рассматриваемого гибридного алгоритма; также представлена реализация параллелизации рассматриваемого алгоритма. В работе рассмотрены две версии параллелизации: распараллеливание всех шагов гибридного алгоритма и распараллеливание муравьиного алгоритма в составе гибридного.

Исследование проводилось на задаче коммивояжера. Задача состоит в нахождении кратчайшего пути, проходящего через все заданные города ровно один раз с конечным возвращением в исходный город. В качестве примера был использован набор данных berlin52.

1. Алгоритм муравьиной колонии для задачи коммивояжера. Задача коммивояжера часто применяется для тестирования алгоритмов комбинаторной оптимизации, в том числе для исследования алгоритма муравьиной колонии [8, 13–14]. Задача формулируется следующим образом: коммивояжеру необходимо посетить каждый из n городов ровно один раз и вернуться в исходный город за минимальное количество потраченного времени, или преодолев наименьшее суммарное расстояние [14]. В терминах теории графов задача может быть сформулирована как нахождение кратчайшего гамильтонова цикла.

Задачу относят к классу NP, поэтому поиск точного решения задачи не всегда возможен из-за ограничений по времени и вычислительных ресурсов. В этом случае на практике часто используются приближенные, эвристические алгоритмы, находящие субоптимальные решения. Эвристические алгоритмы различаются как по затрачиваемым ресурсам (время, память), так и по точности найденного решения.

Алгоритм муравьиной колонии, относящийся к метаэвристическому классу роевых алгоритмов [15], имитирует поведение колонии добывающих пищу муравьев.

В живой природе муравьи разыскивают пищу не направлено — случайно и хаотично. Когда пища найдена, муравьи приносят ее в муравейник. Возвращаясь, они оставляют следы с феромонами, которые служат сигналом для других муравьев. Если другие муравьи найдут путь с феромонами, они, вероятнее всего, выберут его. Если муравьи доберутся до источника пищи, они также начнут оставлять феромоны на обратном пути. Со временем феромон начинает испаряться, поэтому найденные пути становятся менее притягательными для муравьев. Чем короче путь, тем меньше феромона успевает испариться во время

прохождения муравья от источника пищи до муравейника: прохождение по короткому пути затрачивает меньше времени в то время, как плотность феромонов остается высокой.

Испарение играет важную роль в поиске глобальных экстремумов: если феромоны испаряются с низкой скоростью, путь, найденный алгоритмом, с высокой вероятностью может представлять локальный экстремум [16]. С другой стороны, высокая скорость испарения не позволяет муравьям эксплуатировать найденные решения, превращая поиск в случайный. Таким образом, когда один муравей находит оптимальный (или субоптимальный) путь от источника пищи до муравейника, другие муравьи с большей вероятностью выберут тот же путь, что в конечном итоге приведет всех или большую часть муравьев к одному и тому же пути.

2. Генетический алгоритм. Генетический алгоритм относится к классу метаэвристических алгоритмов, который моделирует естественный отбор. Также как и муравьиный алгоритм, генетический алгоритм часто используется для задач комбинаторной оптимизации. Он использует принципы и терминологию, заимствованные из генетики. Также как и муравьиный алгоритм, генетический алгоритм образует класс схожих алгоритмов, использующих одну идею моделирования естественного отбора [17]. В таких алгоритмах потенциальное решение некоторой проблемы называют особью, которая кодируется особым образом (в простейшем случае — двоичным числом). Совокупность особей, являющихся потенциальными решениями, называется популяцией.

Поиск оптимального решения задачи осуществляется в последовательном преобразовании одного конечного набора решений в другой с использованием генетических операторов выбора (селекции), скрещивания (кроссинговера) и мутации.

Оператор селекции выбирает особей, образуя новую популяцию, для последующего применения остальных операторов. Как правило, выбираются особи, решение которых наиболее близко к оптимальному. Существует множество способов реализации оператора селекции [18].

Оператор кроссинговера создает новую популяцию особей с помощью попарного преобразования особей текущей популяции. Он объединяет некоторым образом информацию двух особей для создания новой. Как и оператор выбора, оператор скрещивания может быть реализован различными способами. Этот оператор имеет свою собственную реализацию в зависимости от типа данных, с которыми работает алгоритм [19].

Оператор мутации позволяет получать принципиально новых особей. Обычно оператор мутации с небольшой вероятностью изменяет некоторую часть особи-решения [18].

Преимущество алгоритма заключается в том, что эволюционная идея может быть применена в разных задачах и разными способами [20].

3. Гибридный алгоритм муравьиной колонии с изменяющимися с помощью генетического алгоритма параметрами. Рассматриваемый алгоритм впервые представлен в работе [12]. Основанный на работе двух метаэвристических алгоритмов, данный метод является гибридным. Основное отличие алгоритма от подобных заключается в том, что он не является последовательным применением нескольких алгоритмов. Его идея состоит в том, чтобы применить генетический алгоритм для оптимизации параметров основного алгоритма оптимизации — муравьиной колонии.

Алгоритм имеет динаические параметры и статические. Для каждого муравья a_k определены следующие динамические параметры α_k, β_k, Q_k.

α_k — чувствительность к феромонам.

β_k — чувствительность муравья к эвристической информации.

Q_k — интенсивность феромона. Определяет количество феромона, которое будет отложено муравьем.

Таким образом, в отличие от классической реализации, в данной модификации предполагается, что перечисленные параметры не применяются ко всем муравьям одновременно, а напротив, каждый муравей имеет свой независимый набор параметров.

Алгоритм содержит и статические общие параметры: n, ρ, τ_0.

n — количество муравьев.

τ_0 — параметр, определяющий количество феромона на дугах графа при инициализации.

ρ — скорость испарения феромона.

Каждый муравей k на итерации t выбирает дугу ij с вероятностью $p_{ij}^k(t)$, определяемую формулой

$$p_{ij}^k(t) = \frac{\tau_{ij}^{\alpha_k}(t)\eta_{ij}^{\beta_k}(t)}{\sum \tau_{iu}^{\alpha_k}(t)\eta_{iu}^{\beta_k}(t)}. \tag{1}$$

Каждый муравей откладывает феромон $\Delta\tau_{ij}$ в соответствии с формулой:

$$\Delta\tau_{ij} = \frac{Q_k}{f(x_k)}, \tag{2}$$

где $f(x_k)$ — длина пути x, найденная муравьем k.

Над динамическими параметрами работают генетические операторы. Оператор селекции, который вероятностно определяет новый набор параметров, задается следующим образом:

$$P(M_k) = \frac{f(x_k)}{\sum f(x_k)}, \tag{3}$$

где $P(M_k)$ — вероятность выбора k-го параметра муравья для дальнейшего его участия в работе алгоритма.

Оператором кроссинговера является побитовое сложение (исключающее «или») битовых представлений значений параметров. Оператор мутации случайным образом изменяет бит битового представления каждого параметра.

Последовательность действий представлена в алгоритме 1.

Алгоритм 1. Алгоритм муравьиной колонии с изменяющимися с помощью генетического алгоритма параметрами.

$c = \inf$
$x = \varnothing$
define n, τ_0, ρ
for each $v_{ij} \in V$ **do**
    $\tau_{ij} \sim (0; \tau_0)$
end for
for each $a_k \in A$ **do**
    $\alpha_k = \alpha, \beta_k = \beta, Q_k = Q$
end for
while stopping criteria is not reached **do**
    for each $a_k \in A$ **do**
        $x_k(t) = \varnothing$
        while $|x_k| \neq N$ **do**

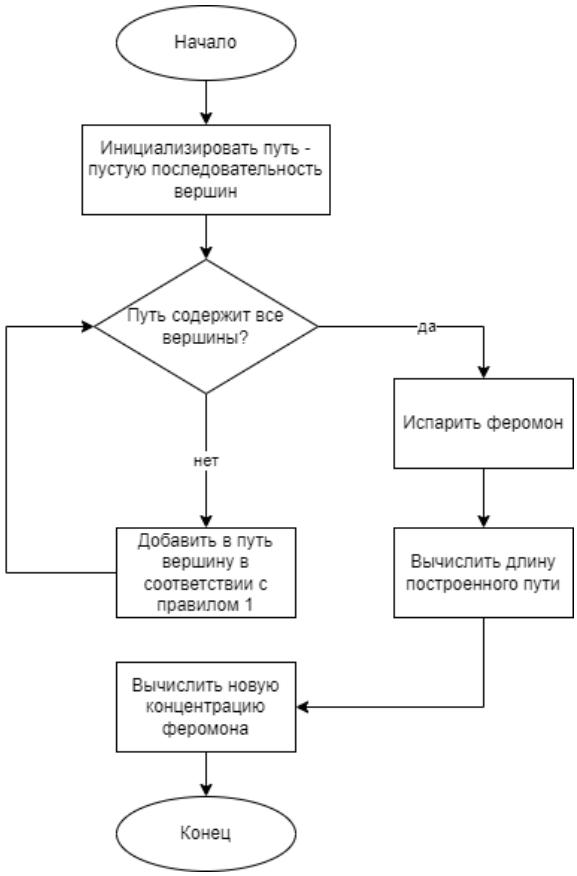

Рис. 1. Блок-схема работы муравья в гибридном алгоритме

choose j according to the rule (1)
$x_k(t) = x_k(t) \cup \{(i, j)\}$
end while
if $f(x_k(t)) < c$ **then**
$c = f(x_k(t))$
$x = x_k(t)$
end if
for each $v_{ij} \in x_k$ **do**
calculate $\Delta\tau_{ij}$ according to the rule (2)
end for
end for
Apply selection operator according to the rule (3)
Apply crossover operator
Apply mutation operator
Set new parameters α_k, β_k, Q_k
end while

4. Параллельная реализация. Известно, что муравьиные алгоритмы имеют параллельную реализацию [7], так как поиск решения отдельного муравья одной итерации не зависит от результатов поиска других муравьев. Однако рассматриваемый метод является гибридным, а значит, существует возможность появления новых зависимостей, не поддающихся распараллеливанию.

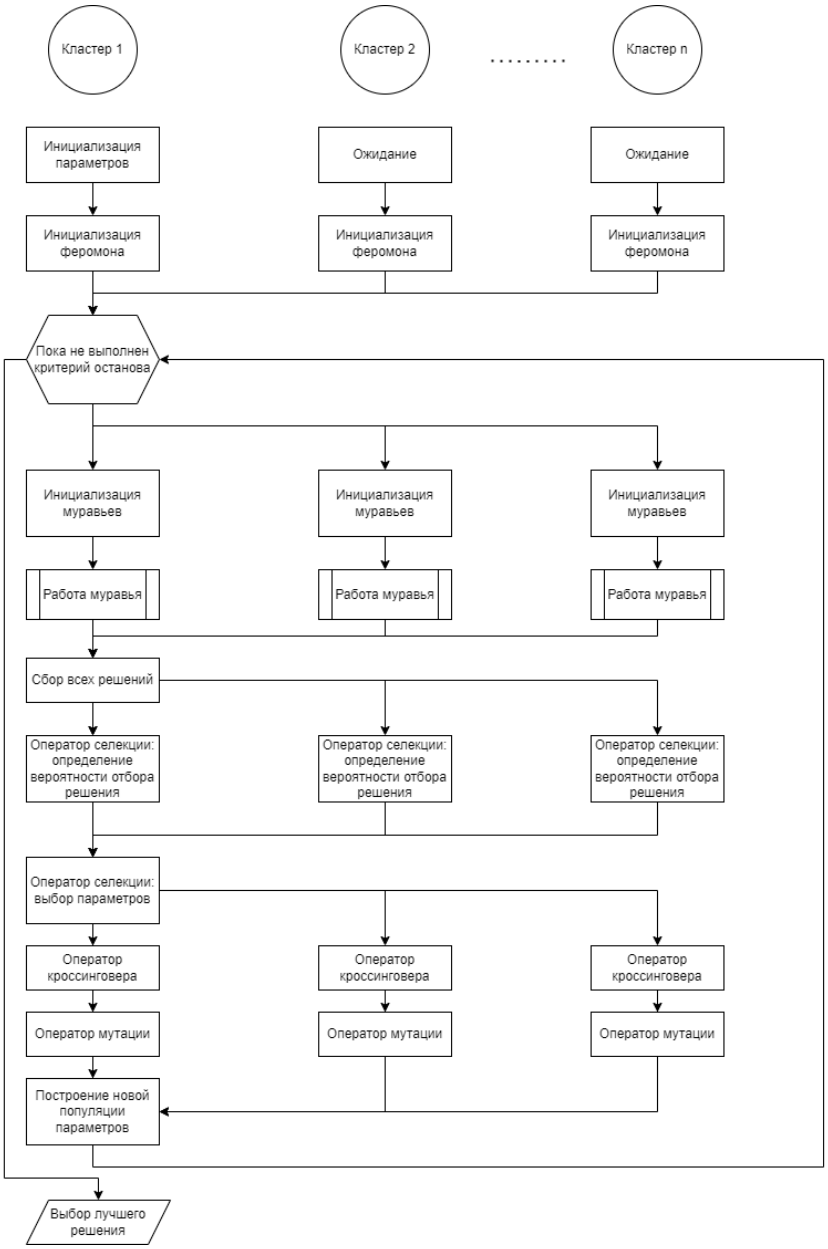

Рис. 2. Схема работы параллельной реализации гибридного алгоритма

Генетический алгоритм также имеет параллельную реализацию. Генетический алгоритм может быть распараллелен двумя способами. Первый заключается в том, чтобы разделить исходную популяцию на количество вычислительных кластеров. Затем к каждой из полученных популяций необходимо последовательно применить генетические операторы. Этот способ не подходит, так как количество муравьев обычно невелико по сравнению с количеством вершин графа [10], и новые популяции будут содержать небольшое количество особей, что приводит к неэффективному использованию оператора кроссинговера.

Второй способ заключается в распараллеливании работы каждого оператора по отдельности. Но при таком подходе все вычислительные кластеры вынуждены ожидать

завершение вычислений остальных кластеров при выполнении одного из генетических операторов.

Из вышесказанного можно предположить, что параллелизация генетической части гибридного алгоритма не является эффективной. Поэтому предложены два варианта распараллеливания: полное распараллеливание и распараллеливание без генетической части. Последовательность действий полного распараллеливания представлена в алгоритме 2.

Алгоритм 2. Параллельная модификация алгоритма муравьиной колонии с изменяющимися с помощью генетического алгоритма параметрами.

$c = \inf$
$x = \varnothing$
define n, τ_0, ρ
for each $p \in P$ **do**
  $V_p = \{v_{ij}\}; V_p \in V$
end for
for each $p \in P$ **do**
  for each $v_{ij} \in V_p$ **do**
    $\tau_{ij} \sim (0; \tau_0)$
  end for
end for
for each $a_k \in A$ **do**
  $\alpha_k = \alpha, \beta_k = \beta, Q_k = Q$
end for
for each $p \in P$ **do**
  $A_p = \{a_k\}; A_p \in A$
end for
while stopping criteria is not reached **do**
  for each $p \in P$ **do**
    for each $a_k \in A_p$ **do**
      $x_k(t) = \varnothing$
      while $|x_k| \neq N$ **do**
        choose j according to the rule (1)
        $x_k(t) = x_k(t) \cup \{(i, j)\}$
      end while
      if $f(x_k(t)) < c$ **then**
        $c = f(x_k(t))$
        $x = x_k(t)$
      end if
      for each $v_{ij} \in x_k$ **do**
        calculate $\Delta\tau_{ij}$ according to the rule (2)
      end for
    end for
  end for
  for each $p \in P$ **do**
    Apply selection operator according to the rule (3)
  end for
  Define pairs of parameters α_k, β_k, Q_k
  for each $p \in P$ **do**
    Apply crossover operator

Таблица 1

Время выполнения реализации гибридного алгоритма с полным распараллеливанием в зависимости от количества вычислительных потоков

Количество потоков	Худшее время выполнения алгоритма, с	Лучшее время выполнения алгоритма, с	Среднее время выполнения алгоритма, с
1	9.7454	9.1452	9.5773
2	5.4874	5.0037	5.3414
3	4.0908	3.5210	3.9144
4	3.3602	3.2449	3.2449

Таблица 2

Время выполнения реализации гибридного алгоритма с распараллеливанием муравьиного алгоритма в зависимости от количества вычислительных потоков

Количество потоков	Худшее время выполнения алгоритма, с	Лучшее время выполнения алгоритма, с	Среднее время выполнения алгоритма, с
1	9.6271	8.9387	9.3635
2	5.5467	5.0636	5.3888
3	4.1952	3.7568	4.0610
4	3.6997	3.0645	3.3789

Apply mutation operator
end for
Set new parameters α_k, β_k, Q_k
end while

Здесь P — множество вычислительных кластеров, A — множество муравьев, V — множество ребер графа.

Схема алгоритма работы каждого отдельно взятого муравья представлена на рис. 1.

Схема параллельной реализации представлена на рис. 2.

В работе не представлена схема распараллеливания алгоритма вторым способом (без распараллеливания генетической части алгоритма), так как она во многом повторяет распараллеливание первым способом с тем отличием, что все генетические операторы выполняются на одном вычислительном кластере.

5. Результаты. Алгоритм реализован на языке программирования Python3. Для распараллеливания процессов использовался программный интерфейс OpenMP.

Алгоритм был протестирован с параметрами: $\alpha_k = 0.2$, $\beta_k = 1$, $Q_k = 15$, $n = 12$, $\tau_0 = 0.01$, $\rho = 0.9$

Критерием останова является количество итераций: 100, так как в работе [12] показано, что при заданном количестве итераций на данных berlin52 в среднем алгоритм начинает сходиться. При заданных параметрах алгоритм запускался 50 раз для каждого из числа потоков: 1, 2, 3, 4.

В табл. 1 представлено время выполнения программы при различном количестве вычислительных потоков.

Исходя из данных табл. 1, можно сделать вывод, что распараллеливание рассматриваемого алгоритма увеличивает скорость его выполнения.

В табл. 2 представлено время выполнения алгоритма с распараллеливанием муравьиного алгоритма, но без распараллеливания генетического алгоритма.

Исходя из данных табл. 1 и 2, можно сделать вывод, что распараллеливание генетической части рассматриваемого гибридного алгоритма не имеет сильного влияния на время выполнения. Объяснением может являться тот факт, что количество вычислений, которые проводятся генетическим алгоритмом, сравнительно меньше, чем количество вычислений, проводимых муравьиным. Из этого можно заключить, что распараллеливание генетической части рассматриваемой модификации не является целесообразным.

Заключение. В работе рассмотрены возможность и реализация распараллеливания гибридного алгоритма муравьиной колонии с изменяющимися с помощью генетического алгоритма параметрами. Показано, что распараллеливание алгоритма ускоряет его работу. Выяснено, что распараллеливание генетических операторов в рассматриваемом алгоритме не вносит значительного ускорения, поэтому достаточно реализовать параллельную реализацию работы муравьиного алгоритма.

Список литературы

1. Zhang N. Moore's law is dead, long live moore's law! // arXiv preprint arXiv:2205.15011, 2022.

2. Pujol R., Jorba J., Tabani H., Kosmidis L., Mezzetti E., Abella J., Cazorla F. Vector extensions in cots processors to increase guaranteed performance in real-time systems, 2022. ACM Transactions on Embedded Computing Systems (TECS).

3. Zhou K. Feng X. A collaborative grouping aggregation query scheme on heterogeneous computing systems // in 2022 7th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA), 2022. P. 53–61.

4. Regassa D., Yeom H., Son Y. Harvesting the aggregate computing power of commodity computers for supercomputing applications // Applied Sciences, 2022. V. 12, N 10, P. 5113.

5. Ong B. W., Schroder J. B. Applications of time parallelization // Computing and Visualization in Science, 2020. V. 23, N 1. P. 1–15.

6. Blagoveshchenskaya E., Zuev D., Kuznecova I., Tihomirov S. Prilozheniya algoritmov pryamyh razlozhenij abelevyh grupp bez krucheniya k zadacham rasparallelivaniya vychislitel'nyh i konstruktivnyh processov // in Mezhdunarodnye Kolmogorovskie chteniya-XIV, posvyashchennye 100-letiyu professora Z. A. Skopeca, 2017. P. 38–40.

7. Stutzle T. Parallelization strategies for ant colony optimization // in International Conference on Parallel Problem Solving from Nature. Springer, 1998. P. 722–731.

8. Hassan M., Hasan M., Hashem M. et al. An improved acs algorithm for the solutions of larger tsp problems. arXiv preprint arXiv:1304.3763, 2013.

9. Liang D., Zhan Z.-H., Zhang Y., Zhang J. An efficient ant colony system approach for new energy vehicle dispatch problem // IEEE Transactions on Intelligent Transportation Systems, 2019. V. 21. N 11. P. 4784–4797.

10. Zhou X., Ma H., Gu J., Chen H., Deng W. Parameter adaptation-based ant colony optimization with dynamic hybrid mechanism // Engineering Applications of Artificial Intelligence, 2022. V. 114. P. 105139.

11. Olivas F., Valdez F., Castillo O., Gonzalez C. I., Martinez G., Melin P. Ant colony optimization with dynamic parameter adaptation based on interval type-2 fuzzy logic systems // Applied Soft Computing, 2017. V. 53. P. 74–87.

12. Blagoveshchenskaya E. A., Mikulik I. I., Strëungmann L. H. Ant colony optimization with parameter update using a genetic algorithm for travelling salesman problem // Models and Methods for Researching Information Systems in Transport 2020 (MMRIST 2020), 2020. N 1. P. 20–25.

13. Baydogmus G. K. A parallelization based ant colony optimization for travelling salesman problem // in 2022 1st International Conference on Information System & Information Technology (ICISIT). IEEE, 2022. P. 166–169.

14. Liu G., Xu X., Wang F., Tang Y. Solving traveling salesman problems based on artificial cooperative search algorithm // Computational Intelligence and Neuroscience, 2022. V. 2022.

15. El-Khatib S. , Skobtsov Y. , Rodzin S. , Zakharov V. Comparison of modified object detected graph cut and hybrid ant colony optimization-k-means for mri images segmentation // in Computer Science On-line Conference. Springer, 2021. P. 701–708.

16. Brand M., Masuda M., Wehner N., Yu X.-H. Ant colony optimization algorithm for robot path planning // in 2010 international conference on computer design and applications, V. 3. IEEE, 2010. P. V3–436.

17. Whitley D. Next generation genetic algorithms: a user's guide and tutorial // in Handbook of metaheuristics. Springer, 2019. P. 245–274.

18. Katoch S., Chauhan S. S., Kumar V. A review on genetic algorithm: past, present, and future // Multimedia Tools and Applications, 2021. V. 80, N 5. P. 8091–8126.

19. Singh G., Gupta N. A study of crossover operators in genetic algorithms // in Frontiers in Nature-Inspired Industrial Optimization. Springer, 2022. P. 17–32.

20. Lambora A., Gupta K., Chopra K. Genetic algorithm-a literature review // in 2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon). IEEE, 2019. P. 380–384.

Микулик Илья Игоревич — аспирант кафедры «Высшая математика» Петербургского государственного университета путей сообщения Императора Александра I, e-mail: `mikulik.ilia@gmail.com`. Область профессиональных интересов: прикладная алгебра, программирование, методы и алгоритмы оптимизации, параллельные вычисления. Является автором 5 научных работ.

Ilya Mikulik — postgraduate student of Department of Mathematics at Emperor Alexander I St. Petersburg State Transport University, e-mail: `mikulik.ilia@gmail.com`. His research interests include applied algebra, programming, optimization methods and algorithms, parallel computing. He is the author of 5 papers.

Благовещенская Екатерина Анатольевна — д-р. физ.-мат. наук, проф., заведующий кафедрой «Высшая математика» Петербургского государственного университета путей сообщения Императора Александра I. E-mail: `blagoveschenskaya@pgups.ru`. Область научных интересов: фундаментальная и прикладная алгебра, междисциплинарные связи (музыка, математика, лингвистика), распараллеливание алгоритмов. Является автором монографии «Почти вполне разложимые абелевы группы и их кольца эндоморфизмов» (Изд-во СПбГПУ, 2009) и более 120 научных статей, опубликованных в рецензируемых журналах, в том числе "Communications in algebra", "Lecture notes in pure and applied mathematics", "Contemporary Mathematics".

Ekaterina Blagoveshchenskaya — Dr., Professor, Head of Department of Mathematics at Emperor Alexander I St. Petersburg State Transport University. E-mail: `blagoveschenskaya@pgups.ru`. Her research interests include fundamental and applied algebra, interdisciplinary fields (music, mathematics, linguistics), parallelization of algorithms. She is author of more than 120 papers published in peer-reviewed journals, including "Communications in algebra", "Lecture notes in pure and applied mathematics", "Contemporary Mathematics".

Дата поступления — 31.01.2023