http://dx.doi.org/10.35596/1729-7648-2025-31-4-65-72

УДК 519.852.67

РЕШЕНИЕ ЗАДАЧИ КОММИВОЯЖЕРА БОЛЬШИХ РАЗМЕРОВ В КОНТЕКСТЕ ЛОГИСТИЧЕСКОЙ СТРАТЕГИИ ПРЕДПРИЯТИЯ

Ю. О. ГЕРМАН, Е. А. СЕМИЖОН

Белорусский государственный университет информатики и радиоэлектроники (Минск, Республика Беларусь)

Аннотация. Описан подход, основанный на редукции большой задачи коммивояжера к множеству более простых. Представлен алгоритм решения этой задачи с подробной иллюстрацией и описанием логики выполняемых шагов. Алгоритм допускает повторное посещение городов, что для практических целей приемлемо, поскольку основная задача – поиск маршрута минимальной общей длины с заходом в каждый город как минимум единожды. Новизной является учет внутри- и межкластерных связей через пересчет матрицы попарных кратчайших расстояний между населенными пунктами. При этом исходная сеть населенных пунктов разбивается на кластеры. В пределах каждого кластера отыскивается оптимальный маршрут, проходящий через все его узлы без возврата в стартовый узел. После чего ищется кратчайший маршрут, связывающий кластеры. Учет этого фактора более четко «разводит» пары населенных пунктов в результирующей кластерной структуре, что способствует получению качественных решений.

Ключевые слова: задача коммивояжера большой размерности, эвристический алгоритм решения, кластеры вершин, минимизация межкластерных связей.

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Для цитирования. Герман, Ю. О. Решение задачи коммивояжера больших размеров в контексте логистической стратегии предприятия / Ю. О. Герман, Е. А. Семижон // Цифровая трансформация. 2025. Т. 31, № 4. С. 65–72. http://dx.doi.org/10.35596/1729-7648-2025-31-4-65-72.

SOLUTION OF THE LARGE-SCALE TRAVELING SALESMAN PROBLEM IN THE CONTEXT OF THE COMPANY'S LOGISTICS STRATEGY

YULIYA O. GERMAN, EKATERINA A. SEMIZHON

Belarusian State University of Informatics and Radioelectronics (Minsk, Republic of Belarus)

Abstract. This paper describes an approach based on reducing the large traveling salesman problem to a set of simpler ones. A solution algorithm is presented with a detailed illustration and a description of the logic behind the steps. The algorithm allows for repeated visits to cities, which is acceptable for practical purposes, since the primary objective is to find a route of minimal overall length, visiting each city at least once. A novel feature is the consideration of intra- and inter-cluster connections by recalculating the matrix of pairwise shortest distances between settlements. The initial network of settlements is then divided into clusters. Within each cluster, an optimal route is found that passes through all its nodes without returning to the starting node. The shortest route linking the clusters is then found. Taking this factor into account more clearly “separates” pairs of settlements in the resulting cluster structure, which facilitates obtaining high-quality solutions.

Keywords: large-scale traveling salesman problem, heuristic solution algorithm, vertex clusters, minimization of intercluster connections.

Conflict of interests. The authors declare that there is no conflict of interests.

For citation. German Yu. O., Semizhon E. A. (2025) Solution of the Large-Scale Traveling Salesman Problem in the Context of the Complany's Logistics Strategy. *Digital Transformation.* 31 (4), 65–72. http://dx.doi.org/10.35596/1729-7648-2025-31-4-65-72 (in Russian).

Введение

Одним из возможных подходов к решению задач большой размерности является разбиение объектов задачи на кластеры. В задачах на сетях и графах кластеры могут иметь множество различных входных и выходных узлов. Помимо всего прочего, разбиение на кластеры должно в значительной степени минимизировать число межкластерных связей. В [1] описаны задачи по определению кластеров, их использование. В разных источниках отмечается, что кластеры формируются по принципу близости входящих в них узлов. Несмотря на множество положительных результатов применения точных методов для решения большой задачи коммивояжера (Big Traveling Salesman Problem, BTSP), среднестатистический случай может быть связан с вычислительными трудностями. Так, в [2] рассматривается метод ветвей и границ, который относится к категории точных методов, а в [3] – динамическое программирование. Один из наиболее известных эвристических подходов – стратегия ближайшего соседа [4]. Метод ближайшего соседа может дать решение, которое в худшем случае находится в пределах двукратного расстояния от оптимального решения. Однако, как подчеркивается в [5], жадные алгоритмы могут давать решения, весьма далекие от точных, и во многих случаях непредсказуемы. Заслуживают внимания другие эвристические и метаэвристические методы [6] (метод отжига, генетические алгоритмы, метод муравьиных колоний и др.). Их общая проблема – зацикливание в точках локальных оптимумов. При этом число локальных оптимумов может быть весьма большим, что ведет либо к плохому конечному результату, либо к существенному росту вычислительного времени.

В статье рассмотрена возможность дополнительного учета критерия минимума межкластерных связей, описан подход, основанный на редукции BTSP к множеству более простых. Представлен алгоритм решения BTSP с иллюстрацией и описанием логики выполняемых шагов. В качестве новой разработки – учет внутри- и межкластерных связей через пересчет матрицы попарных кратчайших расстояний между населенными пунктами. Учет этого фактора более четко «разводит» пары населенных пунктов в результирующей кластерной структуре.

Развернутый пример с объяснением метода

В качестве примера решения BTSP рассмотрим граф с вершинами, представляющими города, и дугами, веса которых задают расстояния между городами. Требуется обойти все города, побывав в каждом только один раз, вернуться в исходный город и в общей сложности совершить путь минимальной возможной длины. Задача может включать несколько сотен/тысяч населенных пунктов, поэтому точное решение в силу NP-полноты задачи получить практически нельзя за приемлемое время. Рассмотрим общий принцип редукции размерности на примере рис. 1. На рисунке над ребрами указаны расстояния в условных единицах (у. е.). Отсутствие ребра означает отсутствие дороги, соединяющей соответствующие пункты. Стартовый город соответствует вершине 5.

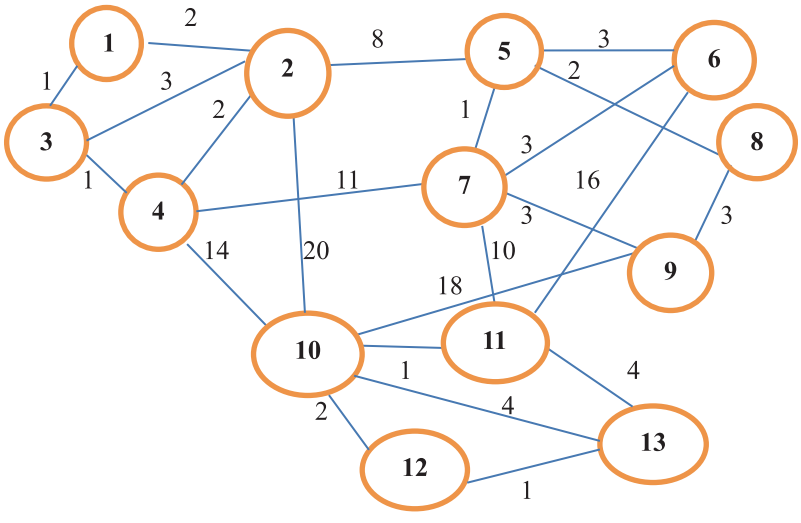

Рис. 1. Исходный граф
Fig. 1. Original graph

Идея редукции состоит в разбиении множества вершин графа на кластеры. Каждый кластер объединяет вершины, попарное расстояние между которыми существенно меньше, чем расстояние между парами вершин из разных кластеров. На первом этапе каждый найденный кластер заменяется одной вершиной – центроидом. Центроиды соединяются ребрами согласно связям в исходном графе. Таким образом, получается новый граф на центроидах, размер которого существенно меньше размера исходного графа, что позволяет сравнительно просто отыскать оптимальное решение на центроидах. На втором этапе центроиды заменяются соответствующими подграфами (кластерами), и оптимальные пути отыскиваются в каждом кластере вершин.

Для отыскания кластеров вершин введем матрицу *D* расстояний между вершинами. Для каждой пары вершин в *D* запишем длину кратчайшего пути, который их связывает. При отыскании кратчайших путей между парами вершин воспользуемся алгоритмом Флойда – Уоршелла [7]. Полученная по этому алгоритму матрица *SD* кратчайших расстояний между каждой парой городов показана на рис. 2.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	0	2	1	2	10	13	11	12	14	16	17	18	19
2	2	0	3	2	8	11	9	10	12	16	17	18	19
3	1	3	0	1	11	14	12	13	15	15	16	17	18
4	2	2	1	0	10	13	11	12	14	14	15	16	17
5	10	8	11	10	0	3	1	2	4	12	11	14	15
6	13	11	14	13	3	0	3	5	6	14	13	16	17
7	11	9	12	11	1	3	0	3	3	11	10	13	14
8	12	10	13	12	2	5	3	0	3	14	13	16	17
9	14	12	15	14	4	6	3	3	0	14	13	16	17
10	16	16	15	14	12	14	11	14	14	0	1	2	4
11	17	17	16	15	11	13	10	13	3	1	0	3	4
12	18	18	17	16	14	16	13	16	16	2	3	0	1
13	19	19	18	17	15	17	14	17	17	4	4	1	0
∑	135	127	136	127	101	128	101	120	121	133	133	150	162

Рис. 2. Матрица кратчайших расстояний *SD*
Fig. 2. The shortest distances matrix *SD*

На матрице *SD* будем отыскивать кластеры. Алгоритм выполняется как итерационная многошаговая процедура. На каждой итерации из матрицы вычеркивается строка и одноименный столбец с максимальной суммой элементов. Процесс вычеркиваний следует прекратить, когда остающиеся не вычеркнутыми элементы в *SD* имеют близкие значения, характерные для кластера вершин. Эти значения можно ориентировочно оценить с помощью метода *K*-средних, если разбить все множество весов ребер на два или три кластера, для чего можно использовать скрипт на языке Python. Программа строит два кластера. Первый кластер К* содержит значения длин ребер [1, 2, 3, 4, 8]. То есть выбираются ребра с наименьшей длиной, или близкой к наименьшей. Второй кластер К** содержит остальные значения. Таким образом, можно считать, что внутрикластерные расстояния определяются первым кластером К*. Имея эти оценки, можно осуществить поиск кластеров.

Определение очередного кластера выполняется на основе процедуры последовательного удаления строк и одноименных столбцов из матрицы *SD* с максимальной текущей суммой элементов. Так, вначале вычеркиваются строка и столбец 13 (с максимальной суммой элементов, равной 162). На следующей итерации вычеркиваются строка и столбец 12 с максимальной суммой элементов, равной 149 (рис. 3). Затем вычеркиваются строка и столбец 10 (с суммой элементов 127) и т. д. Процесс останавливается на матрице, в которой расстояния между парами населенных пунктов не превосходят максимального расстояния для кластера К* (рис. 4).

Первый кластер найден: он представлен не вычеркнутыми строками/столбцами: Cluster 1 = = {1, 2, 3, 4}. Чтобы найти следующий кластер, нужно из матрицы *SD* удалить строки и столбцы кластера Cluster 1, пересчитать веса ребер кластера К* и повторить проделанные выше шаги; опуская их описание, получим второй кластер: Cluster 2 = {5, 6, 7, 8, 9}. Далее по аналогии находим третий кластер: Cluster 3 = {10, 11, 12, 13}. В результате исходный граф можно представить редуцированным графом на кластерах (рис. 5).

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	0	2	1	2	10	13	11	12	14	16	17	18
2	2	0	3	2	8	11	9	10	12	16	17	18
3	1	3	0	1	11	14	12	13	15	15	16	17
4	2	2	1	0	10	13	11	12	14	14	15	16
5	10	8	11	10	0	3	1	2	4	12	11	14
6	13	11	14	13	3	0	3	5	6	14	13	16
7	11	9	12	11	1	3	0	3	3	11	10	13
8	12	10	13	12	2	5	3	0	3	14	13	16
9	14	12	15	14	4	6	3	3	0	14	13	16
10	16	16	15	14	12	14	11	14	14	0	1	2
11	17	17	16	15	11	13	10	13	3	1	0	3
12	18	18	17	16	14	16	13	16	16	2	3	0
∑	116	108	118	110	86	111	87	103	104	129	129	149

Рис. 3. Матрица расстояний после удаления строки и столбца 13
Fig. 3. The distance matrix after deleting row and column 13

	1	**2**	**3**	**4**
1	0	2	1	2
2	2	0	3	2
3	1	3	0	1
4	2	2	1	0

Рис. 4. Определение первого кластера на *SD*
Fig. 4. The first cluster definition on matrix *SD*

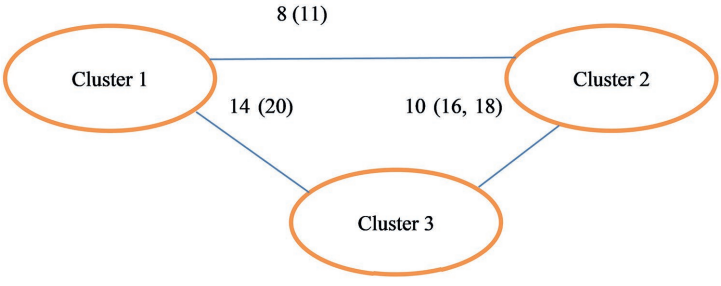

Рис. 5. Исходный граф на кластерах
Fig. 5. Original graph on clusters

Над ребрами (и в скобках) на рис. 5 указаны длины ребер, связывающих кластеры. Гамильтонов путь на этом рисунке – единственный, его минимальная длина: 8 + 10 + 14 = 32. Поскольку стартовая вершина 5, и она принадлежит кластеру 2, то из исходного графа на рис. 1 выберем выход длиной 8 в кластер 1 (рис. 6). Далее находим переход из кластера 1 в кластер 3 (рис. 7).

Очевидно, что переход из кластера 3 в кластер 2 и обход последнего возможны единственным образом, как показано на рис. 8.

Итак, найдено одно из возможных решений (не единственное, впрочем). Его длина составляет 60 у. е. – это оптимальный путь. Для проверки можно использовать программу на Python, реализующую метод ветвей и границ. Выходной скриншот имеет вид, показанный на рис. 9: ответ тот же – 60 у. е.

Рассмотренный подход не свободен от недостатков, поскольку число кластеров велико, и их взаимные связи не позволяют просто отыскать гамильтонов путь. То есть необходимо снова строить кластеры на кластерах.

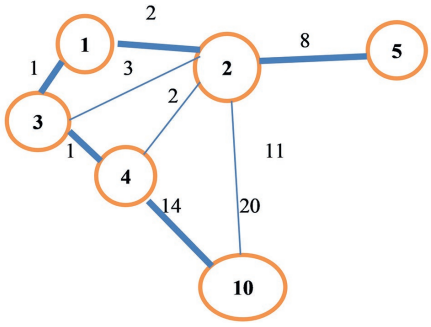

Рис. 6. Переход из кластера 2 в кластер 1 (жирные линии)
Fig. 6. Transition from cluster 2 to cluster 1 (bold lines)

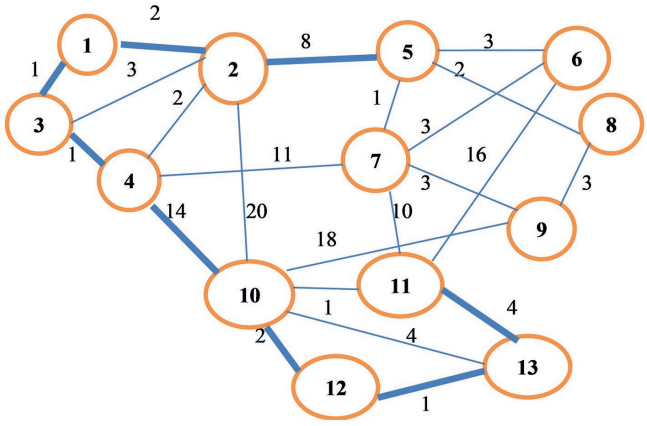

Рис. 7. Переход в кластер 3 и его обход
Fig. 7. Transition to cluster 3 and its traversing

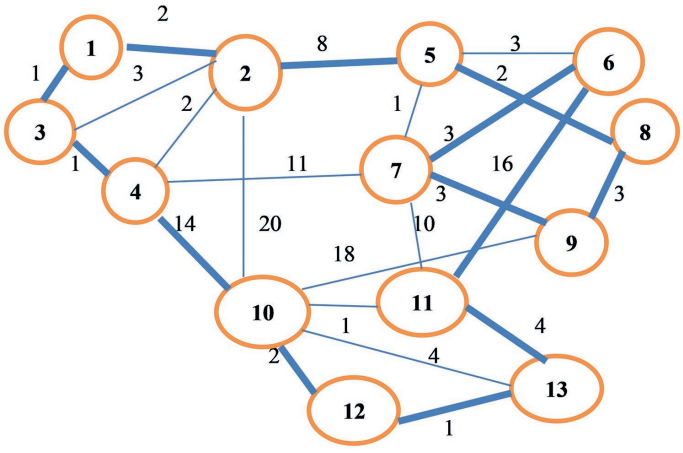

Рис. 8. Переход в кластер 2 и его обход
Fig. 8. Transition to cluster 2 and its traversing

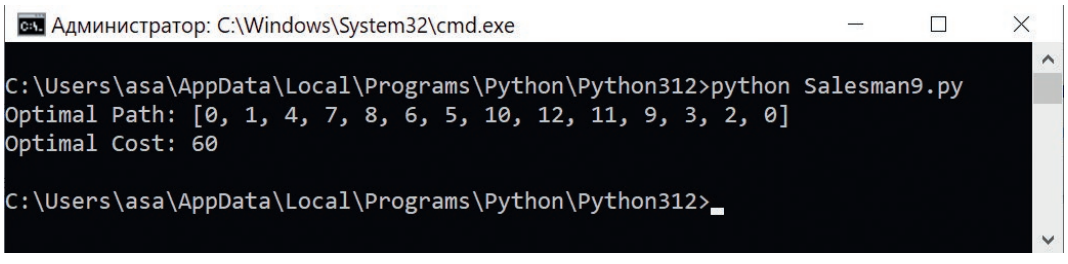

Рис. 9. Оптимальное решение, найденное методом ветвей и границ
Fig. 9. Optimal solution found by branches and boundaries method

Обсуждение и анализ результатов исследований

Качество описанного метода существенно зависит от кластерной структуры исходной сети. Практическая апробация кластерного подхода, как показано в [1], может дать отклонение от точного ответа более чем в два раза (10–30 % в среднестатистическом случае). Четко организованные кластеры можно оценить с помощью метрик, например, метрики силуэта, метрики Девиса – Боулдена и др. [8]. Чем более хаотичен разброс населенных пунктов, тем выше число кластеров и, как отмечено выше, может потребоваться строить кластеры на кластерах. И, наоборот, чем четче кластерная структура, тем выше вероятность получения оптимального итогового решения.

Таким образом, структура кластеров имеет важное значение. Если исходный граф разбить на кластеры {1, 2, 3, 4}, {5, 6, 7, 8, 9, 10}, {11, 12, 13}, то нет точного решения на основе описанной концепции – это первая проблема. Возможный выход из нее – допустимость повторных визитов в ранее посещенные населенные пункты. При этом следует задействовать матрицу парных расстояний, определенную на основе алгоритма Флойда – Уоршелла. Очевидно, и при таком рассмотрении задача остается NP-трудной с классической постановкой в качестве возможного частного варианта. Вторая основная проблема – найти кратчайший маршрут, соединяющий кластеры.

При выполнении исследований максимизировалось число связей в каждом кластере и минимизировалось число связей между кластерами. Для «настройки» алгоритма на эту цель модифицировали матрицу кратчайших расстояний (рис. 2), найденную по методу Флойда – Уоршелла, следующим образом. Если узлы сети были соединены ребром напрямую, то расстояние не изменялось. Если кратчайший путь между парой вершин содержал k ребер, то увеличивали расстояние на значение $(k - 1)\Delta$ (Δ – среднее значение в ранее найденном кластере весов К*, соответствующее рассматриваемой матрице расстояний; на первом этапе К* = [1, 2, 3, 4, 8], т. е. $\Delta = 18/5 = 3{,}6$). Тем самым увеличивалось расстояние между каждой парой не связанных напрямую ребрами узлов, что повысило вероятность их отнесения в разные кластеры.

Матрица расстояний после пересчета имела вид, показанный на рис. 10, где нижняя строка соответствует суммам элементов в столбцах оригинальной матрицы расстояний. Следует отметить, что соотношения расстояний (больше/меньше) в обеих матрицах не всегда сохраняются. В модифицированной матрице расстояний эти соотношения более выражены.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	0	2	1	5,6	13,6	20,2	18,2	19,2	24,8	23,2	27,8	28,8	33,4
2	2	0	3	2	8	14,6	12,6	13,6	15,6	19,6	24,2	25,2	29,8
3	1	3	0	1	18,2	21,2	15,6	20,2	22,2	18,6	23,2	24,2	28,8
4	5,6	2	1	0	13,6	20,2	11	19,2	24,8	14	18,6	19,6	24,2
5	13,6	8	18,2	13,6	0	3	1	2	7,6	19,2	14,6	24,8	22,2
6	20,2	14,6	21,2	20,2	3	0	3	8,6	9,6	21,2	16,6	26,8	24,2
7	18,2	12,6	15,6	11	1	3	0	3	3	14,6	10	20,2	17,6
8	19,2	13,6	20,2	19,2	2	8,6	3	0	3	24,8	20,2	26,8	27,8
9	24,8	15,6	22,2	24,8	7,6	9,6	3	3	0	21,2	16,6	26,8	24,2
10	23,2	19,6	18,6	14	19,2	21,2	14,6	24,8	21,2	0	1	2	4
11	27,8	24,2	23,2	18,6	14,6	16,6	10	20,2	16,6	1	0	6,6	4
12	28,8	25,2	24,2	19,6	24,8	26,8	20,2	26,8	26,8	2	6,6	0	1
13	33,4	29,8	28,8	24,2	22,2	24,2	17,6	27,8	24,2	3	4	1	0
∑	217,8	170,2	197,2	173,8	147,8	189,2	129,8	188,4	199,4	182,4	183,4	232,8	241,2
	135	127	136	127	101	128	101	120	121	133	133	150	162

Рис. 10. Модифицированная матрица кратчайших расстояний *SD*
Fig. 10. The modified shortest distances matrix *SD*

После процедуры удалений с порядком вычеркиваний 13, 12, 1, 4, 2, 10, 11 получаем в итоге кластер Cluster 2 = {5, 6, 7, 8, 9} (рис. 11).

	a5	**a6**	**a7**	**a8**	**a9**
5	0	3	1	2	7,6
6	3	0	3	8,6	9,6
7	1	3	0	3	3
8	2	8,6	3	0	3
9	7,6	9,6	3	3	0
∑	38,4	51	30,2	43,4	50

Рис. 11. Кластер 2 после процедуры удалений
Fig. 11. Cluster 2 after the deletion procedure

Следует отметить, что пороговым расстоянием для включения в кластер являлось значение, смещенное на величину 3,6 (то есть 8 + 3,6 = 11,6). Результат в итоге совпадает с ранее полученным. Очевидно, что отношения расстояний между узлами, попадающими в один и тот же кластер, и разные кластеры в модифицированной версии алгоритма существенно более различимы. Таким образом, характер внутри- и межкластерных связей соответствует поставленной цели.

Заключение

1. Описан подход к решению логистической задачи о коммивояжере большой размерности, который заключается в разбиении множества вершин графа на кластеры (в кластер попадают вершины, попарные расстояния между которыми имеют близкие значения); в поиске оптимального маршрута, проходящего через центры кластеров; в детализации найденного маршрута в каждом кластере. При этом допускается возможность повторного посещения городов, что для практических целей приемлемо, так как основная задача – поиск маршрута минимальной общей длины с заходом в каждый город как минимум единожды.

2. Рассмотрен алгоритм с подробной иллюстрацией и описанием логики выполняемых шагов. Основное новшество – учет внутри- и межкластерных связей через пересчет матрицы попарных кратчайших расстояний между населенными пунктами. Учет этого фактора более четко «разводит» пары населенных пунктов в результирующей кластерной структуре.

3. Представленный подход объединяет точные и приближенные методы и может быть использован для задач коммивояжера большой размерности.

Список литературы / References

1. Karakoyun M. (2019) A New Approach Based on K-Means Clustering and Shuffled Frog Leaping Algorithm to Solve Travelling Salesman Problem. *Proceedings of the 7th International Symposium on Innovative Technologies in Engineering and Science, 22–24 Nov.*
2. Land A. H., Doig A. G. (1960) Optimality and Infeasibility in Combinatorial Programming. *Mathematical Programming*. 1 (1), 122–136.
3. Bellman R. (1962) Dynamic Programming and Lattice Theory. *Proceedings of the National Academy of Science*s. 48 (10), 2010–2015.
4. Croes G. A. (1958) The Traveling Salesman Problem. *Operations Research*. 6 (6), 791–812.
5. *When Greedy Does Not Work – Traveling Salesman*. Available: https://www.neeldhara.com/materials/cpnotes/w03/lec15 (Accessed 5 May 2025).
6. Luke S. (2011) Essentials of Metaheuristics. *Lecture Notes. George Mason University*. USA.
7. Sedgewick R., Wayne K. (2011) Algorithms. *Addison Wesley*. Available: https://algs4.cs.princeton.edu. (Accessed 21 October 2025).
8. Bishop M. C. (2006) Pattern Recognition and Machine Learning. *Springer*.

Поступила 14.07.2025 Принята в печать 17.10.2025 Доступна на сайте 12.01.2026
Received: 14 July 2025 Accepted: 17 October 2025 Available on the website: 12 January 2026

Вклад авторов

Герман Ю. О. сформулировала общую идею работы.
Семижон Е. А. предложила алгоритмическое решение.

Authors' contribution

German Yu. O. formulated the general idea of the work.
Semizhon E. A. proposed an algorithmic solution.

Сведения об авторах

Герман Ю. О., канд. техн. наук, доц. каф. информационных технологий автоматизированных систем, Белорусский государственный университет информатики и радиоэлектроники

Семижон Е. А, ассис. каф. вычислительных методов и программирования, асп., Белорусский государственный университет информатики и радиоэлектроники

Адрес для корреспонденции

220005, Республика Беларусь,
Минск, ул. Платонова, 39
Белорусский государственный университет
информатики и радиоэлектроники
Тел.: +375 17 293-89-04
E-mail: jgerman@bsuir.by
Герман Юлия Олеговна

Information about the authors

German Yu. O., Cand. Sci. (Tech.), Associated Professor at the Department of Information Technologies of Automated Systems, Belarusian State University of Informatics and Radioelectronics

Semizhon E. A., Assistant at the Computational Methods and Programming Department, Postgraduate, Belarusian State University of Informatics and Radioelectronics

Address for correspondence

220005, Republic of Belarus,
Minsk, Platonova St., 39
Belarusian State University
of Informatics and Radioelectronics
Tel.: +375 17 293-89-04
E-mail: jgerman@bsuir.by
German Yuliya Olegovna