2024, **25** (4), 476–482. doi 10.26089/NumMet.v25r436

doi 10.26089/NumMet.v25r436 УДК 004.023; 51–37; 519.178; 519.688

Точное и приближенное решения задачи коммивояжера большого размера

В. В. Бурховецкий
Южный федеральный университет,
Институт математики, механики и компьютерных наук имени И. И. Воровича,
Ростов-на-Дону, Российская Федерация
ORCID: 0000-0002-1292-0280, e-mail: buvictor95@inbox.ru

Б. Я. Штейнберг
Южный федеральный университет,
Институт математики, механики и компьютерных наук имени И. И. Воровича,
Ростов-на-Дону, Российская Федерация
ORCID: 0000-0001-8146-0479, e-mail: borsteinb@mail.ru

Аннотация: В работе рассматривается быстрый эвристический алгоритм для задачи коммивояжера на основе метода ветвей и границ с оценкой качества получаемого решения, т.е. с параметром k, который гарантирует, что получаемое решение хуже оптимального не более чем в $1/k$ раз. Алгоритм предназначен для вычислительных систем с общей памятью.

Ключевые слова: задача коммивояжера, точный алгоритм, эвристический алгоритм, параллельный алгоритм, метод ветвей и границ.

Для цитирования: Бурховецкий В.В., Штейнберг Б.Я. Точное и приближенное решения задачи коммивояжера большого размера // Вычислительные методы и программирование. 2024. **25**, № 4. 476–482. doi 10.26089/NumMet.v25r436.

Exact and approximate solutions of the traveling salesman problem of large size

Victor V. Burkhovetskiy
Southern Federal University,
Vorovich Institute of Mathematics, Mechanics, and Computer Science,
Rostov-on-Don, Russia
ORCID: 0000-0002-1292-0280, e-mail: buvictor95@inbox.ru

Boris Ya. Steinberg
Southern Federal University,
Vorovich Institute of Mathematics, Mechanics, and Computer Science,
Rostov-on-Don, Russia
ORCID: 0000-0001-8146-0479, e-mail: borsteinb@mail.ru

Abstract: This paper describes a fast heuristic algorithm based on branch-and-bound for the traveling salesman problem with a set approximation ratio, i.e. with a parameter k that guarantees that the solution obtained by the algorithm is at most $1/k$ times worse than the optimal solution. The algorithm is designed for shared-memory systems.

2024, **25** (4), 476–482. doi 10.26089/NumMet.v25r436

Keywords: traveling salesman problem, exact algorithm, heuristic algorithm, parallel algorithm, branch-and-bound.

For citation: V. V. Burkhovetskiy, B. Ya. Steinberg, "Exact and approximate solutions of the traveling salesman problem of large size," Numerical Methods and Programming. **25** (4), 476–482 (2024). doi 10.26089/NumMet.v25r436.

1. Введение. В последнее время наблюдается рост множества прикладных задач с дискретными математическими моделями. Среди них много NP-трудных задач, о которых бытует мнение, что для их решения можно применять только эвристические алгоритмы, а точные методы — нельзя. Среди NP-трудных задач наиболее популярна задача коммивояжера, причем к этой задаче большого размера сводится задача сборки генома de novo [1].

Задача коммивояжера (замкнутая) — задача нахождения гамильтонова цикла с минимальной суммой дуг в полном взвешенном ориентированном графе, стоимости дуг которого неотрицательны. К этой задаче сводится также задача коммивояжера на неориентированном графе: из любого неориентированного графа можно получить ориентированный, заменив в нем каждое ребро стоимости c на две противоположно направленные дуги, каждая из которых имеет стоимость c. Поэтому в дальнейшем рассматриваются только полные ориентированные графы. К замкнутой задаче коммивояжера сводится разомкнутая задача коммивояжера — задача поиска гамильтонова пути минимальной стоимости — пути, проходящего через каждую вершину в точности один раз. Для такой сводимости достаточно к исходному графу добавить одну вершину, которая соединяется со всеми другими вершинами дугами веса 0: после нахождения в новом графе минимального гамильтонова цикла добавленная вершина удаляется вместе с инцидентными к ней дугами.

В основе предлагаемых алгоритмов лежит модифицированный алгоритм Балаша–Кристофидеса. Алгоритм Балаша–Кристофидеса [2] — точный алгоритм решения задачи коммивояжера, основанный на методе ветвей и границ ("точный" означает, что он всегда находит оптимальное решение). Модификация алгоритма Балаша–Кристофидеса (1978 г.) состоит в адаптации этого алгоритма к современным процессорам. В частности, алгоритм был распараллелен и был уменьшен расход памяти, в результате чего он стал работать быстрее Concorde [3], как описано в работе [4]. Новизна данной работы, по сравнению с [4], заключается в том, что алгоритм был сделан эвристическим и в него был добавлен параметр k, который гарантирует, что получаемое решение хуже оптимального не более чем в $1/k$ раз. Целью работы является получить возможность решать задачи больших размеров.

Распараллеливание метода ветвей и границ на распределенной памяти для задачи о ранце представлено в [5]. Обход дерева вариантов в методе ветвей и границ можно описывать рекурсивной функцией. Распараллеливание рекурсивных функций рассматривалось в [4, 6]. В работе [7] рассмотрены параллельные (векторные) алгоритмы решения некоторых других задач теории графов: поиск в ширину, поиск всех пар кратчайших путей в графе от заданной вершины, ранжирование вершин, поиск связных компонент. В [8] рассмотрен полиномиальный эвристический алгоритм решения задачи коммивояжера с оценкой качества найденного решения. Вычислительная сложность этого алгоритма $O(n^3)$, где n — число вершин графа, а найденное этим алгоритмом решение хуже оптимального не более чем в 1.5 раза.

Поиск экстремумов функций с использованием дерева решений применяется в задачах как дискретной, так и непрерывной оптимизации. При обходе дерева решений возникают избыточные вычисления. Анализ влияния избыточных вычислений на время выполнения проводился в работе [9], хотя, очевидно, такое влияние очень сильно зависит от алгоритма и от вычислительной архитектуры (особенно для параллельных алгоритмов).

Представленный в данной работе алгоритм является развитием [4, 10].

Программная реализация точного алгоритма доступна по ссылке [11].

2. Особенности распараллеливания метода ветвей и границ. Приведем известные элементы алгоритмов, основанных на методе ветвей и границ, для анализа возможностей и особенностей распараллеливания.

Метод ветвей и границ применяется к построению точных алгоритмов решения оптимизационных переборных задач большого размера. Разнообразие прикладных задач такого типа влечет различия алгоритмов их решения. Тем не менее в многообразии этих алгоритмов можно выделить общие особенности.

2024, **25** (4), 476–482. doi 10.26089/NumMet.v25r436

Алгоритмы, основанные на методе ветвей и границ, имеют две основные составные части: 1) построение и обход дерева вариантов; 2) правила отсечения неперспективных ветвей дерева вариантов.

Иногда для одной и той же задачи можно построить несколько различных деревьев вариантов и различные правила отсечения ветвей.

Построение и обход дерева вариантов применяется не только к задачам оптимизации (в основном дискретной), но и к задачам пересмотра всех элементов большого конечного множества, описанного некоторыми комбинаторными условиями, например при составлении списка всех клик (или максимально независимых множеств) графа [12]. Распараллеливание обхода дерева вариантов может применяться и к таким задачам.

Будем обозначать через f целевую функцию задачи, минимум которой необходимо найти на большой конечной области определения. Будем полагать, что множество листов дерева вариантов является областью определения функции f (так происходит во многих задачах рассматриваемого типа). Дерево вариантов содержит все точки области определения целевой функции.

При обходе дерева вариантов при достижении алгоритмом листа можно вычислить значение целевой функции. Лучшее на текущий момент работы алгоритма вычисленное значение целевой функции f будем называть рекордным значением и обозначать f_{rec}.

Будем полагать, что в узлах дерева вариантов задана некоторая функция F, обладающая следующими свойствами:

1) для любого узла N дерева вариантов и любого листа L на ветке этого узла выполняется неравенство $F(N) \leqslant f(L)$;
2) для любого узла N дерева вариантов и любого предшествующего ему узла P (т.е. узла, лежащего на пути от корня дерева до данного узла) выполняется неравенство $F(N) \geqslant F(P)$.

Такую функцию F называют оценкой целевой функции f. Если для целевой функции задана оценка, то во многих случаях на листах дерева значение оценки равно значению целевой функции: $F(L) = f(L)$. Если на дереве вариантов задана оценка целевой функции, то можно использовать *правило отсечения ветвей*: если выполняется

$$F(N) \geqslant f_{\text{rec}}, \tag{1}$$

то на ветке узла N не находится оптимальное значение целевой функции и эту ветку можно не рассматривать алгоритмом (отсечь).

Как обычно, распараллеливание метода ветвей и границ заключается в одновременном выполнении обработки ветвей дерева вариантов разными вычислительными потоками. При этом некоторые особенности основанных на методе ветвей и границ алгоритмов отрицательно влияют на эффективность распараллеливания:

1) разное время обработки разных ветвей дерева вариантов может создавать плохой загрузочный баланс;
2) текущее рекордное значение целевой функции является общей обновляемой переменной для всех потоков.

3. Некоторые особенности алгоритма Балаша–Кристофидеса. В основанном на методе ветвей и границ алгоритме Балаша–Кристофидеса [2] оценка целевой функции строится на основе итерационного венгерского алгоритма решения вспомогательной задачи о назначениях. Балаш и Кристофидес модифицировали венгерский алгоритм — они разработали способ найти решение задачи о назначениях в узле дерева вариантов на основе ее решения в отце этого узла. По их словам, для этого “почти всегда достаточно одной итерации венгерского алгоритма”. У корня дерева вариантов нет узла-отца, а поэтому решение задачи о назначениях нужно находить итерационным венгерским алгоритмом без модификации Балаша и Кристофидеса. Как покажут представленные в данной статье численные эксперименты, решение вспомогательной задачи о назначениях в корне дерева вариантов дает оценку целевой функции, очень близкую в процентном отношении к оптимальному точному решению задачи коммивояжера.

Исходные графы в статье [2] представлены матрицей весов дуг. Для численных экспериментов веса дуг берутся как случайные целые числа в диапазоне от 0 до 1000.

4. Ускорение алгоритма за счет уменьшения точности. В этом параграфе представлены эвристические алгоритмы, основанные на точном алгоритме ветвей и границ.

Как и в большинстве алгоритмов, основанных на методе ветвей и границ, в рассматриваемом алгоритме строится дерево вариантов, в каждом узле дерева вариантов вычисляется оценка целевой функции и сравнивается с текущим рекордным значением целевой функции.

2024, **25** (4), 476–482. doi 10.26089/NumMet.v25r436

Если условие (1) истинно, то ветка дерева вариантов данного узла отсекается, поскольку на ней не может быть значений целевой функции, лучших текущего рекорда. Отличие представленного в данной статье алгоритма от алгоритма [10, 13] в том, что вместо условия (1) стоит условие

$$F(N) \geqslant k \cdot f_{\text{rec}} \tag{2}$$

с заранее заданным коэффициентом k, где $0 < k \leqslant 1$.

Если коэффициент k равен 1, то получается алгоритм [10, 13], который находит точное решение. Для случая $k < 1$ о результате работы можно сказать, что он отличается от оптимального не более чем в $1/k$ раз. Например, если $k = 0.9$, то найденный алгоритмом с условием (2) результат хуже оптимального не более чем на 10%; если $k = 0.5$, то найденный алгоритмом с условием (2) результат хуже оптимального не более чем в 2 раза.

5. Уменьшение расхода памяти с помощью ссылочной структуры. Еще одной проблемой, препятствующей решению задач больших размеров, был чрезмерный расход памяти. Он возникал ввиду того, что на каждом узле дерева вариантов создавался новый граф, представленный в виде матрицы весов. Из-за этого ранее не получалось решать задачи с матрицами весов размера выше 3000. Для решения этой проблемы авторы представили матрицу весов в виде ссылочной структуры данных, описанной в работе [4], которая хранит только отличия графа на данном узле дерева вариантов от графа в узле-отце. Использование этой структуры позволило уменьшить расход памяти в сотни раз, как видно из табл. 1.

Таблица 1. Расход памяти параллельным алгоритмом до и после внедрения структуры

Table 1. Memory usage by the parallel algorithm before and after integrating the structure

Размер матриц Matrix size	Максимальное использование памяти, МБ Maximum memory usage, MB	
	До Before	После After
1000	1 668	25
1500	5 235	50
2000	11 351	82
2500	9 575	97
3000	22 394	149
5000	—	361
7000	—	1 431
10000	—	1 530

6. Условия проведения численных экспериментов. Представленные в данной работе результаты были получены в следующих условиях:

- процессор Intel Core i5-12400
  - 6 ядер;
  - с hyperthreading;
  - L3 кэш: 18 МБ (общий);
  - L2 кэш: 1280 КБ (у каждого ядра свой);
  - L1 кэш: 32 КБ для инструкций, 48 КБ для данных (у каждого ядра свой);
- операционная система Debian 11 (Linux);
- язык программирования C++;
- компилятор GCC версии 10.2.1 с флагами `-Ofast` и `-march=native`;
- GNU OpenMP v. 10.2.1;
- оперативная память DDR5, 32 ГБ, частота 5600 МГц;
- функция измерения времени: `omp_get_wtime` для OpenMP, `clock_gettime` для pthread.

Все измерения времени работы и используемой памяти проводились на матрицах со случайными целыми числами диапазона от 0 до 1000000, которые хранились в целых 4-байтных переменных. Суммы элементов матриц (например, рекорд, оценка целевой функции) хранились в целых 8-байтных переменных. Всего для численных экспериментов было сгенерировано по 5 случайных матриц каждого размера.

2024, **25** (4), 476–482. doi 10.26089/NumMet.v25r436

7. Результаты численных экспериментов. Сначала проводился эксперимент, целью которого было понять, какие значения параметра k можно задавать, чтобы сильно ускорить алгоритм, но при этом потерять как можно меньше точности. Для этого на разных размерах запускался точный ($k = 1$) алгоритм и вычислялось:

1) насколько близким к оптимальному является решение начальной задачи о назначениях (в корне дерева вариантов);
2) сколько раз меняется значение рекорда;
3) насколько рекорд улучшается каждый раз.

Результаты представлены в табл. 2.

С ростом размера случайного исходного графа значение начального рекорда приближается к оптимальному значению целевой функции и для размера 7000 улучшение начального рекорда составляет менее 0.02%. Одной из причин этого является значительный вклад в целевую функцию оценки, которая получена в начале алгоритма с использованием решения задачи о назначениях.

В результате было выбрано значение параметра $k = 0.95$.

Целью следующего эксперимента было замерить время работы эвристического алгоритма на задачах больших размеров, на которых точный алгоритм работает слишком долго (больше 4 часов на задачах размера 20000). Для этого использовался только эвристический алгоритм с коэффициентом $k = 0.95$, т.е. о найденном результате можно говорить, что он хуже оптимального не более чем на 5%. При работе алгоритма на задачах размера 40000 ему достаточно 32 ГБ оперативной памяти, а на задачах размера 50000 уже нет — он начинает использовать swap (swap находится на NVMe SSD диске, поэтому даже с использованием swap время получается не слишком большим). Результаты представлены в табл. 3.

Таблица 2. Среднее улучшение начального рекорда при работе алгоритма

Table 2. Average improvement of the initial record during runtime

Размер матриц Matrix size	Среднее отставание начального решения от оптимального Average difference between the initial and the optimal solution	Среднее количество изменений значения рекорда Average number of record changes	Диапазон улучшений рекорда Range of record improvements
1000	0.1534%	3.80	0.0005%–0.2155%
2000	0.0591%	3.80	0.0001%–0.0988%
3000	0.0681%	5.20	0.0004%–0.0834%
5000	0.0319%	4.60	0.0001%–0.0504%
7000	0.0212%	5.00	0.0001%–0.0193%
10000	0.0136%	4.60	0.0001%–0.0172%

Таблица 3. Среднее время выполнения и расход памяти алгоритма для графов большого размера

Table 3. Average running time and memory usage of the algorithm on large graphs

Размер матриц Matrix size	Среднее время Average time	Максимальное использование памяти Maximum memory usage
20000	52 с 52 s	6.2 ГБ 6.2 GB
30000	3 мин 23 с 3 min 23 s	14.3 ГБ 14.3 GB
40000	4 мин 47 с 4 min 47 s	24.7 ГБ 24.7 GB
50000	13 мин 58 с 13 min 58 s	37.7 ГБ (30.5 ГБ + 7.2 ГБ swap) 37.7 GB (30.5 GB + 7.2 GB swap)

2024, **25** (4), 476–482. doi 10.26089/NumMet.v25r436

8. Направление дальнейших исследований и заключение. В дальнейшем предполагается увеличивать размер рассматриваемых задач, находя в приемлемое время решение, которое "хуже оптимального не более чем на k процентов". Для этого предполагается рассмотреть следующие варианты достижения данной цели:

1) хранить в памяти (в виде списка смежности или списка дуг) только дуги графа, величина которых не более заранее заданного числа s или хранить ровно q дуг наименьшего веса;
2) проводить эксперименты с использованием swap, находящегося на NVMe SSD;
3) рассмотреть распараллеливание алгоритма на высокопроизводительном кластере (с распределенной памятью).

Список литературы

1. *Miller J.R., Koren S., Sutton G.* Assembly algorithms for next-generation sequencing data // Genomics. 2010. **95**, N 6. 315–327. doi 10.1016/j.ygeno.2010.03.001.
2. *Balas E., Christofides N.* A restricted Lagrangean approach to the traveling salesman problem // Mathematical Programming. 1981. **21**, N 1. 19–46. doi 10.1007/BF01584228.
3. Concorde TSP Solver. http://www.math.uwaterloo.ca/tsp/concorde.html. (Дата обращения: 25 ноября 2024).
4. *Бурховецкий В.В.* Оптимизация и распараллеливание упрощенного алгоритма Балаша–Кристофидеса для задачи коммивояжера // Программные системы: теория и приложения. 2020. **11**, № 4. 3–16. doi 10.25209/2079-3316-2020-11-4-3-16. http://psta.psiras.ru/read/psta2020_4_3-16.pdf. (Дата обращения: 25 ноября 2024).
5. *Посыпкин М.А., Сигал И.Х.* Комбинированный параллельный алгоритм решения задачи о ранце // Известия РАН. Теория и системы управления. 2008. № 4. 50–58.
6. *Feautrier P.* A parallelization framework for recursive tree programs // Lecture Notes in Computer Science. Vol. 1470. Heidelberg: Springer, 1998. 470–479. doi 10.1007/BFb0057890.
7. *Личманов Д.И., Афанасьев И.В., Воеводин В.В.* Исследование производительности архитектурно-независимого фреймворка VGL для эффективной реализации графовых алгоритмов // Вычислительные методы и программирование. 2023. **24**, № 4. 485–499. https://doi.org/10.26089/NumMet.v24r433. (Дата обращения: 25 ноября 2024).
8. *Christofides N.* Worst-case analysis of a new heuristic for the travelling salesman problem // Operations Research Forum. 2022. **3**. Article Number 20. doi 10.1007/s43069-021-00101-z.
9. *Баркалов К.А., Рябов В.В., Сидоров С.В.* О некоторых способах балансировки локального и глобального поиска в параллельных алгоритмах глобальной оптимизации. Вычислительные методы и программирование. 2010. **11**, № 4. 382–387. http://mi.mathnet.ru/vmp333. (Дата обращения: 25 ноября 2024).
10. *Burkhovetskiy V., Steinberg B.* An exact parallel algorithm for traveling salesman problem. https://dl.acm.org/doi/10.1145/3166094.3166108. (Дата обращения: 25 ноября 2024).
11. *Бурховецкий В.В.* Программа точного решения задачи коммивояжера, основанная на модифицированном алгоритме Балаша–Кристофидеса. https://ops.rsu.ru/download/progs/BalasChristofides_v1_0.zip. (Дата обращения: 29 ноября 2024).
12. *Кристофидес Н.* Теория графов. Алгоритмический подход. М.: Мир, 1978.
13. *Бурховецкий В.В.* Оптимизация и распараллеливание упрощенного алгоритма Балаша–Кристофидеса для задачи коммивояжера. Ростов-на-Дону: Инст. Матем. Механ. Компьют. наук, 2018. http://2018.nscf.ru/TesisAll/4Students/237_BurkhovetskiyVV.pdf. (Дата обращения: 25 ноября 2024).

Поступила в редакцию
26 сентября 2024 г.

Принята к публикации
20 ноября 2024 г.

Информация об авторах

Виктор Витальевич Бурховецкий — аспирант; Южный федеральный университет, Институт математики, механики и компьютерных наук имени И. И. Воровича, ул. Мильчакова, 8а, 344090, Ростов-на-Дону, Российская Федерация.

Борис Яковлевич Штейнберг — д.т.н., старший научный сотрудник; Южный федеральный университет, Институт математики, механики и компьютерных наук имени И. И. Воровича, ул. Мильчакова, 8а, 344090, Ростов-на-Дону, Российская Федерация.

2024, **25** (4), 476–482. doi 10.26089/NumMet.v25r436

References

1. J. R. Miller, S. Koren, and G. Sutton, "Assembly Algorithms for Next-Generation Sequencing Data," Genomics **95** (6), 315–327 (2010). doi 10.1016/j.ygeno.2010.03.001.
2. E. Balas and N. Christofides, "A Restricted Lagrangean Approach to the Traveling Salesman Problem," Math. Program. **21** (1), 19–46 (1981). doi 10.1007/BF01584228.
3. Concorde TSP Solver. http://www.math.uwaterloo.ca/tsp/concorde.html. (Cited November 25, 2024).
4. V. V. Burkhovetskiy, "Optimization and Parallelization of the Simplified Balas' and Christofides' Algorithm for the Traveling Salesman Problem," Program Systems: Theory and Applications **11** (4), 3–16 (2020). doi 10.25209/2079-3316-2020-11-4-3-16. http://psta.psiras.ru/read/psta2020_4_3-16.pdf. (Cited November 25, 2024).
5. M. A. Posypkin and I. Kh. Sigal, "A Combined Parallel Algorithm for Solving the Knapsack Problem," Izv. Akad. Nauk, Teor. Sist. Upravl., No. 4, 50–58 (2008) [J. Comput. Syst. Sci. Int. **47** (4), 543–551 (2008)]. doi 10.1134/S1064230708040072.
6. P. Feautrier, "A Parallelization Framework for Recursive Tree Programs," in *Lecture Notes in Computer Science* (Springer, Heidelberg, 1998), Vol. 1470, pp. 470–479. doi 10.1007/BFb0057890.
7. D. I. Lichmanov, I. V. Afanasyev, and Vad. V. Voevodin, "Performance Study of the Architecture-Independent VGL Framework for Efficient Implementation of Graph Algorithms," Numerical Methods and Programming **24** (4), 485–499 (2023). https://doi.org/10.26089/NumMet.v24r433. (Cited November 25, 2024).
8. N. Christofides, "Worst-Case Analysis of a New Heuristic for the Travelling Salesman Problem," Oper. Res. Forum **3**, Article Number 20 (2022). doi 10.1007/s43069-021-00101-z.
9. K. A. Barkalov, V. V. Ryabov, and S. V. Sidorov, "Some Local and Global Search Balancing Methods in Parallel Global Optimization Algorithms," Numerical Methods and Programming **11** (4), 382–387 (2010). http://mi.mathnet.ru/vmp333. (Cited November 25, 2024).
10. V. Burkhovetskiy and B. Steinberg, "An Exact Parallel Algorithm for Traveling Salesman Problem," https://dl.acm.org/doi/10.1145/3166094.3166108. (Cited November 25, 2024).
11. V. V. Burkhovetskiy, "Implementation of the Exact Algorithm for the Traveling Salesman Problem Based on Modified Balas and Christofides Algorithm," https://ops.rsu.ru/download/progs/BalasChristofides_v1_0.zip. (Cited November 29, 2024).
12. N. Christofides, *Graph Theory: An Algorithmic Approach* (Academic Press, Cambridge, 1975; Mir, Moscow, 1978).
13. V. V. Burkhovetskiy, *Optimization and Parallelization of the Simplified Balas' and Christofides' Algorithm for the Traveling Salesman Problem* (Inst. Matem. Mekh. Komp'yut. Nauk, Rostov-on-Don, 2018). http://2018.nscf.ru/TesisAll/4Students/237_BurkhovetskiyVV.pdf. (Cited November 25, 2024) [in Russian].

Received
September 26, 2024

Accepted for publication
November 20, 2024

Information about the authors

Victor V. Burkhovetskiy — Postgraduate Student; Southern Federal University, Vorovich Institute of Mathematics, Mechanics, and Computer Science, ulitsa Milchakova, 8a, 344090, Rostov-on-Don, Russia.

Boris Ya. Steinberg — Dr. Sci., Senior Scientist; Southern Federal University, Vorovich Institute of Mathematics, Mechanics, and Computer Science, ulitsa Milchakova, 8a, 344090, Rostov-on-Don, Russia.