12. *Kumar N., Karambir, Kumar R.* A comparative analysis of PMX, CX and OX Crossover operators for solving traveling salesman problem, *International Journal of Latest Research in Science and Technology*, 2012, pp. 98-101.
13. *Kureychik V.M., Lebedev B.K., Lebedev O.K.* Poiskovaya adaptatsiya: teoriya i praktika [Search adaptation: theory and practice]. Moscow: Fizmatlit, 2006, 272 p. ISBN 5-9221-0749-6.
14. *Abdoun O., Abouchakaba J, Tajani C.* Analyzing the Performance of Mutation Operators to Solve the Travelling Salesman Problem, *CoRR*, Vol. abs/1203.3099, 2012, pp. 66-77.
15. Parametry i klassy protokolov marshrutizatsii [The parameters and classes of routing protocols]. Available at: http://skif.bas-net.by/bsuir/base/node360.html (Accessed 23 February 2015).
16. TSPLIB. Available at: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95 /tsp/ (Accessed 23 February 2015).
17. *Ciba M., Sekaj I.* Ant colony optimization with re-initialization, *Automation, Control and Intelligent Systems*, 2013, No. 1 (3), pp. 53-66.
18. *Dai Q., Junzhong J., Chunnian L.*, An effective initialization strategy of pheromone for ant colony optimization, *Bio-Inspired Computing*, 2009.
19. *Zhu Q.B., Yang Z.J.* An Ant Colony Optimization Algorithm Based on Mutation and Dynamic Pheromone Updating, *Journal of Software*, 2004, No. 2 (15), pp. 185-192.
20. *Solomon M.M.* Algorithms for the vehicle routing and scheduling problems with time windows constraints, *Operations Research*, 1987, 35, pp. 254-265.

Статью рекомендовал к опубликованию д.т.н., профессор Я.Е. Ромм.

Мартынов Артём Владимирович – Южный федеральный университет; e-mail: sir.aspex@gmail.com; 347928, г. Таганрог, пер. Некрасовский, 44; тел.: 89508459429; кафедра дискретной математики и методов оптимизации; аспирант.

Курейчик Виктор Михайлович – e-mail: kur@tsure.ru; тел.: 89282132730; кафедра дискретной математики и методов оптимизации; зав. кафедрой; д.т.н.; профессор.

Martinov Artem Vladimirovich – Southern Federal University; e-mail: sir.aspex@gmail.com; 44, Nekrasovskiy, Taganrog, 347928, Russia; phone: +79508459429; the department of discrete mathematics and optimization methods; postgraduate student.

Kureichik Viktor Mihaylovich – e-mail: kur@tsure.ru; phone: +79282132730; the department of discrete mathematics and optimization methods; head of department; dr. of eng. sc.; professor.

УДК 681.3.06: 681.323(519.6)

Я.Е. Ромм, Е.Г. Назарьянц

ПОЛИНОМИАЛЬНАЯ СЛОЖНОСТЬ ПАРАЛЛЕЛЬНОЙ ФОРМЫ МЕТОДА ВЕТВЕЙ И ГРАНИЦ РЕШЕНИЯ ЗАДАЧИ КОММИВОЯЖЕРА

Работа содержит параллельное преобразование алгоритма Дж. Литтла реализации метода ветвей и границ для решения задачи коммивояжера на основе идентификации экстремумов при помощи максимально параллельной сортировки подсчетом по матрицам сравнений. Приводится описание и программа сортировки с оценкой временной сложности. Описан метод и реализующие его программные операторы идентификации локальных экстремумов. Предложенный параллельный алгоритм цикличен, даны две оценки его временной сложности $T\left(n^4/6-n^3/4\right)=O\left(n\log_2 n\right)$ *и* $T\left(n^4/6-n^3/4\right)=O\left(n^5\log_2 n\right)$ *для случаев без возвратов к оборванным ветвям и с возвратом к одной из них (без учета вложений). При одновременной обработке всех обрываемых ветвей без учета вложений временная сложность по сравнению с обработкой одной оборванной ветви не увеличивается за счет роста числа процессоров, и имеет место оценка* $T\left(n^6/6\right)=O\left(n^5\log_2 n\right)$*. Оценки с учетом числа процессоров используют абстрактную модель неветвящихся параллельных программ, при этом не учитывается архитектура параллельной вычислительной системы и время обмена. Вы-*

полнено численное моделирование предложенного параллельного алгоритма с помощью его последовательной реализации на персональном компьютере. Представлены результаты численного эксперимента, которые подтверждают правильность работы предложенной параллельной модификации алгоритма Дж. Литтла. Дано сравнение полученных результатов с известными результатами применения различных алгоритмов и программ параллельной реализации метода ветвей и границ для решения задачи коммивояжёра. Существенным отличием предложенного способа решения задачи коммивояжера от известных является объединение параллельной сортировки и собственно метода ветвей и границ, позволяющее получить полиномиальные оценки временной сложности алгоритма.

Задача коммивояжера; параллельная форма метода ветвей и границ; полиномиальная оценка временной сложности; параллельная сортировка подсчетом.

Ya.E. Romm, E.G. Nazaryants

POLYNOMIAL COMPLEXITY OF THE PARALLEL FORM OF THE BRANCH AND BORDERS METHOD FOR SOLVING THE COMMIS VOYAGEUR'S PROBLEM

The work contains the parallel transformation of George Little algorithm of the implementation of branch and borders method for solving the commis voyageur's problem based on the identification of extremums using maximum parallel sorting by counting on comparisons' matrixes. There are the description and the program of sorting with the estimation of time complexity. The method and programmatic operators of the local extremums identification, that are realizing it, are described in the work. The proposed parallel algorithm is cyclical, there are given two estimations of its time complexity $T\left(n^4/6-n^3/4\right)=O\left(n\log_2 n\right)$ *for* $T\left(n^4/6-n^3/4\right)=O\left(n^5\log_2 n\right)$ *for cases without returning to the ragged branches and with returning to one of them (excluding investments). In the simultaneous processing of all ragging branches excluding investment, the time complexity in the comparing with treating of one ragged branch does not increase due to growth of the processors' number, and we have the estimation* $T\left(n^6/6\right)=O\left(n^5\log_2 n\right)$. *Estimates based on the number of processors have been used an abstract model of straight-line parallel programs, wherein it does not take into account the architecture of the parallel computing system and the exchange's time. The numerical simulation of the proposed parallel algorithm is done with using its consistent implementation on a personal computer. The results of numerical experiments which confirm the correct operation of the proposed parallel George Little algorithm's modification are introduced. There are given the comparison of the results with the known results of applying different algorithms and the programs of parallel implementation of the branch and borders method for solving the commis voyageur's problem. The essential difference between the proposed method of solving the commis voyageur's problem from the known methods is a association of parallel sorting and strictly speaking method of branches and borders, providing to get an estimation of polynomial time complexity of algorithm.*

The commis voyageur's problem; a parallel form of the branch and borders method; the polynomial estimation of time complexity of algorithm; parallel sorting by counting.

Введение и постановка вопроса. Задача коммивояжёра (Travelling salesman problem, TSP) – одна из самых известных задач комбинаторной оптимизации, заключающаяся в отыскании наиболее выгодного маршрута, проходящего через указанные города хотя бы по одному разу с последующим возвратом в исходный город. Оптимизационная постановка задачи относится к классу NP-трудных задач. Задача коммивояжера – важная задача транспортной логистики, криптографии. Актуальные аспекты задачи исследуются в [1, 2], а также в [3, 4], традиционная постановка излагается в [5, 6]. Согласно [5, 6] коммивояжер должен выйти из первого города, посетить по одному разу в неизвестном порядке города $2, 3, 4, \ldots, n$ и вернуться в первый город. Расстояния между всеми городами известны. В каком порядке следует обходить города, чтобы замкнутый путь коммивояжера был кратчайшим? Среди городов вводится база – город, где начинаются и заканчиваются

все пути коммивояжёров. Стоимость переезда из города i в город j будет задаваться матрицей, где нужное значение будет храниться на пересечении i-й строки и j-го столбца. На вход конструируемого алгоритма поступает матрица расстояний между n городами и множество значений расстояний от каждого города до базы. Необходимо найти k замкнутых маршрутов, где $k>1$, со следующими свойствами: через каждый город должен проходить один и только один маршрут, причём только один раз; все маршруты должны проходить через базу; для любой пары полученных маршрутов количество городов в них не должно отличаться больше чем на единицу. Суммарная длина полученных маршрутов должна быть как можно меньше.

Параллельный алгоритм строится на основе метода ветвей и границ следующим образом. Задается матрица:

	x_1	x_2	…	x_n
x_1	∞	a_{ij}	…	a_{ij}
x_2	a_{ij}	∞	…	a_{ij}
…	…	…	…	…
x_n	a_{ij}	a_{ij}	…	∞

(1)

где x_1, x_2 ,..., x_n – вершины; $a_{ij} \geq 0$ – длина (или цена) маршрута; $1 \leq i, j \leq n$;

∞ – обозначает путь из вершины x_k в x_k, $1 \leq k \leq n$.

Следуя [5, 6], решение задачи коммивояжера можно разбить на следующие три этапа, которые удобны для распараллеливания [7]:

1. Нахождение нижней границы длин всевозможных маршрутов. Для ее нахождения используется следующее соображение: к элементам произвольно выбранной строки или столбца матрицы задачи коммивояжера можно прибавить или вычесть некоторое число, при этом оптимальность решения не изменится, если вычтенную или прибавленную величину учесть при расчете нижней границы. При этом длина любого маршрута Вычтем из каждой строки число, равное ее минимальному элементу ($a_i = \min(j) a_{ij}$). Вычтем из каждого столбца число, равное минимальному элементу этого столбца ($a_j = \min(i) a_{ij}$). Полученная матрица называется приведенной по строкам и столбцам. Сумма всех вычтенных чисел $f = \sum_{i=1}^{n} a_i + \sum_{j=1}^{n} a_j$ называется константой приведения. Константу приведения следует выбирать в качестве нижней границы длины маршрутов [5, 6].

2. Разбиение множества маршрутов на подмножества. Для выделения претендентов коммивояжера изменится на алгебраическую сумму соответствующих индексам этого маршрута добавленных (вычтенных) величин.

На включение во множество дуг, по которым производится ветвление, рассмотрим в уже приведенной матрице все элементы равные нулю. В строке и в столбце, на пересечении которых находится рассматриваемый ноль с индексами i, j, осуществляется поиск минимальных элементов, не совпадающих с нулем на пересечении. Степень нулевого элемента Θ_{ij} определяется как сумма минимального элемента в соответствующей строке i и минимального элемента в соответствующем столбце j: $\Theta_{ij} = \min(i) a_{ij} + \min(j) a_{ij}$.

После определения всех степеней Θ_{ij} нулевых элементов этой матрицы фиксируется маршрут, которому принадлежат дуги с максимальной степенью нуля, этот маршрут с наибольшей вероятностью (предположительно) является искомым.

3. Исключение маршрутов. Для полученной матрицы маршрутов, включающей дугу (i, j), вычеркиваем в матрице строку i и столбец j, а чтобы не допустить образования цикла в маршруте, заменяем элемент, замыкающий текущую цепочку, на бесконечность (∞), этот элемент симметричен элементу с индексами (i, j) относительно главной диагонали и идентифицируется парой индексов (j, i). Множество всех маршрутов, не включающих дугу (i, j), получаем путем замены элемента a_{ji} на бесконечность.

В заключение выполняется проверка маршрутов на минимальность, и повтор предыдущих этапов, до тех пор, пока не будет найден минимальный маршрут.

Задачей излагаемого сообщения является синтезировать параллельные алгоритмы реализации метода ветвей и границ для задачи коммивояжера с полиномиальными оценками временной сложности. Временная сложность (кратко – время) $T(R)$, где R – количество процессоров, будет измеряться количеством последовательных шагов алгоритма на модели неветвящихся параллельных программ без учета обмена [8, 9]. Синтезируемый алгоритм требуется верифицировать путем численного моделирования на персональном компьютере.

Применяемая сортировка. Используется устойчивая [10] внутренняя сортировка одномерного массива по ключу (ниже – сортировка) с взаимно однозначным соответствием входных и выходных индексов на основе матрицы сравнений (МС) [11, 12], частным случаем которой является модифицированная сортировка подсчетом, излагаемая ниже. Пусть по отношению «$\le$» требуется упорядочить элементы массива

$$\{ c_l \}_{l=1}^{n}, \qquad n < \infty . \tag{2}$$

Массив (1) располагается горизонтально над таблицей и вертикально слева от нее, МС принимает вид [12]:

	c_1	c_2	…	c_n
c_1	α_{11}	α_{12}	…	α_{1n}
c_2	α_{21}	α_{22}	…	α_{2n}
…	…	…	…	…
c_n	α_{n1}	α_{n2}	…	α_{nn}

(3)

где

$$\alpha_{ij} = \begin{cases} 1, & c_i < c_j, \\ 0, & c_i = c_j, \\ -1, & c_j < c_i. \end{cases} \tag{4}$$

$i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n$. Значения α_{ij} из (4) удобно заменить символами "+", " 0 ", " – ". Например, для массива $C = (6.1, 2.2, 7.6, 2.2, 7.3)$ МС (3) примет вид:

	6.1	2.2	7.6	2.2	7.3
6.1	0	-	+	-	+
2.2	+	0	+	0	+
7.6	-	-	0	-	-
2.2	+	0	+	0	+
7.3	-	-	+	-	0

Чтобы определить номер в отсортированном массиве j-го элемента входного массива, достаточно подсчитать число нулей и плюсов в j-м столбце над диагональю, включая диагональный элемент, и сложить это число с числом плюсов j-й строки ниже диагонали. В обозначении отсортированного массива $c1$ получится:

$$c1[3]=c[1],\ c1[1]=c[2],\ c1[5]=c[3],\ c1[2]=c[4],\ c1[4]=c[5].$$

Процедура $sort00$ данной сортировки (Delphi):

```
procedure sort00(var c,c1: vect; var e: vect1);
var i,j,k: integer;
begin
for j:=1 to n do  begin k:= 0;
for i:=1 to j  do if c[j]>=c[i]  then k:=k+1;
for i:=j+1 to n do if c[j]>c[i]  then k:=k+1;
c1[k]:=c[j]; e[k]:=j;
end;  end;
```

Все сравнения в (3) взаимно независимы и инвертируют знак относительно диагонали – сортировка устойчива (сохраняет порядок равных элементов), максимально параллельна с временной сложностью $T(N^2/2)=O(1)$ [11]. Для идентификации экстремальных элементов существенно, что при каждом индексе k элемента $c1[k]$ отсортированного массива, в процедуре $sort00$ выполнено $c1[k]:=c[j]$, и запоминается индекс этого же элемента во входном массиве: $e[k]:=j$, на выходе сортировки $c1[k]$ и $e[k]$ имеют равный индекс k. При каждом k условие локализации (идентификации) минимального элемента в ε-окрестности элемента проверяется в виде системы неравенств

$$\left| e[k]-e[k-L] \right|>\varepsilon,\ L=1,2,..,k-1. \qquad (5)$$

Выполнение одновременно всех неравенств (5) означает, что внутри ε-окрестности входного индекса $e[k]$ элемента массива $c[k]$, нет входного индекса элемента меньшего по значению чем $c1[k]$:

```
k:=1; while k<=n do begin for L:=1 to k-1 do if abs(e[k-L]-e[k]) <=
eps0 then goto 22; writeln(' ',c1[k],' ',e[k]);
 22: k:=k+1 end;
```

В этом фрагменте n из (2), $eps0=\varepsilon$. При условии, что $eps0$ меньше половины расстояния между ближайшими друг к другу локально минимальными элементами, идентифицируются одновременно все локально минимальные элементы в произвольно заданных окрестностях радиуса $eps0$ [12]. Аналогичные идентификации максимума и глобальных экстремумов изложены в [12].

Применение идентификации экстремумов на основе сортировки для параллельного выполнения метода ветвей и границ. Как отмечалось, временная сложность рассматриваемой сортировки в максимально параллельной форме составляет $T(n^2/2)=O(1)$. Операции (5) взаимно независимы по всем индексам, порядок и значения которых готовы после сортировки. Поэтому временная сложность идентификации всех минимумов составит

$$T(R)=O(1), \qquad (6)$$

где согласно (5)

$$R=\sum_{k=1}^{n}\sum_{\ell=1}^{k-1}\ell=\frac{1}{2}\sum_{k=1}^{n}\left(k^2-k\right).$$

Отсюда

$$R \sim \frac{n^3}{6} - \frac{n^2}{4}. \tag{7}$$

С учетом временной сложности предварительной сортировки, время и число процессоров которой не превышают (6), (7), окончательно получим

$$T\left(\frac{n^3}{6} - \frac{n^2}{4}\right) = O(1). \tag{8}$$

Применение (8) к 1-му этапу нахождения нижней границы длин всевозможных маршрутов даст глобальный минимум в каждой строке, затем в каждом столбце матрицы (1). Глобальный минимум одномерного массива находится аналогично (5) с изменением [12]:

$$\left| e[k] - e[k-L] \right| \geq 1, \quad L = 1, 2, \ldots, k-1. \tag{9}$$

Для (9) сохраняются оценки (6), (7). Однако рассматриваемый поиск всех минимумов выполняется не в одном, а сразу в n одномерных массивах (в n строках матрицы (1)). Поэтому число процессоров возрастает в n раз, получится $R \sim \frac{n^4}{6} - \frac{n^3}{4}$, и оценка (8) перейдет в оценку вида:

$$T\left(\frac{n^4}{6} - \frac{n^3}{4}\right) = O(1). \tag{10}$$

Аналогично, с оценкой (10) выполнится поиск минимумов одновременно во всех столбцах матрицы (1). При этом рассматриваемая параллельная обработка столбцов может быть выполнена только после обработки строк, поскольку вычитание в каждой строке своего минимального элемента, вообще говоря, изменит местоположение и значения априорных минимумов в столбцах. По величине порядка оценка (10) не изменится ни для поиска минимумов в столбцах, ни для выполнения первого этапа в целом, на величину порядка не повлияет операция одновременного вычитания минимальных элементов строк из всех элементов этих же строк. В заключение 1-го этапа вычисляется нижняя граница длины полного пути коммивояжера по формуле (1), которая в рассматриваемых обозначениях содержит сложение $2n$ готовых значений слагаемых. Такая сумма может быть вычислена параллельно по схеме сдваивания за время $T(n) = O(\log_2 2n)$, что эквивалентно

$$T(n) = O(\log_2 n). \tag{11}$$

Суммирование (10), (11) влечет заключительную оценку временной сложности 1-го этапа:

$$T\left(\frac{n^4}{6} - \frac{n^3}{4}\right) = O\left(\log_2 n\right). \tag{12}$$

На 2-м этапе выполняется разбиение множества маршрутов на два подмножества: подмножество, включающее дугу (i, j), и подмножество, не включающую эту дугу в искомый маршрут. С этой целью находятся все нули матрицы, полученной на 1-м этапе, путем одновременного сравнения всех элементов матрицы с нулевыми значениями, достаточно n^2 процессоров, чтобы все нули по значению и по индексам идентифицировать за время $T(n^2) = O(1)$. Для каждого нуля находится его степень Θ_{ij}, которая определяется как сумма минимального элемента в соответствующей строке i и минимального элемента в соответствующем столбце j:

$\Theta_{ij} = \min(i)a_{ij} + \min(j)a_{ij}$, при этом не берется в расчет сам нулевой элемент с индексами i, j. Таким образом, временную сложность рассматриваемой операции 2-го этапа можно оценить сверху по соотношению $T\left(n^2\right) = O(1)$, при этом существенно, что размерность n следует заменить размерностью n-1. Именно этим будет определяться сходимость метода и окончательная верхняя оценка сложности:

$$T\left(\frac{(n-1)^4}{6} - \frac{(n-1)^3}{4}\right) = O\left(\log_2 n\right). \tag{13}$$

Найденные степени всех нулей сравниваются путем сортировки, выбирается наибольшая (в конце отсортированного массива), соответствующая ей дуга рассматривается как искомая. Временная сложность рассмотренных преобразований, если не требуется возврата к оборванной ветви, оценивается с учетом уменьшения размерности матрицы на единицу (до матрицы 2×2) на каждом этапе данного вида (всего $n-1$ этапов). Правые части оценок будут иметь такой же вид, как в (12), (13). Левые части оценок аналогичны левым частям (12), (13), но продолжают убывать по размерности на единицу на каждом этапе до размера матрицы 2×2. Отсюда в случае отсутствия ветвлений с соответственными возвратами временная сложность метода ветвей и границ оценивается из соотношения.

$$T\left(\frac{n^4}{6} - \frac{n^3}{4}\right) = O\left(n\log_2 n\right). \tag{14}$$

В левой части (14) проставлено максимальное число процессоров, требуемое на первом этапе.

Если возникнет необходимость возврата к уже отброшенной ветви (найденная минимальная граница больше, чем минимальная граница уже отброшенной ветви на любом из предыдущих этапов), то в худшем случае мы можем вернуться на первый этап, где из исходной матрицы исключен один и только один элемент, который исключаются из дальнейшего рассмотрения. Это будет происходить при каждом возврате в худшем случае, пока в матрице не останется 4 элемента (матрица размером 2×2). Отсюда оценка временной сложности параллельного выполнения метода ветвей и границ для худшего случая примет $T\left(\frac{n^4}{6} - \frac{n^3}{4}\right) = O\left(\sum_{\ell=1}^{n^2-4}\left(n^2-\ell\right)n\log_2 n\right)$, или, $T\left(\frac{n^4}{6} - \frac{n^3}{4}\right) = O\left(\frac{\left(n^2-3\right)\left(n^2-4\right)}{2}n\log_2 n\right)$.

Окончательно,

$$T\left(\frac{n^4}{6} - \frac{n^3}{4}\right) = O\left(n^5\log_2 n\right). \tag{15}$$

Таким образом, в лучшем случае имеет место (14), в худшем – (15).

Теорема 1. В случае отсутствия ветвлений с соответственными возвратами временная сложность предложенной параллельной модификации метода ветвей и границ оценивается из (14). При необходимости возврата к одной отброшенной ветви в худшем случае имеет место оценка (15).

Случай возврата ко всем оборванным ветвям. Если в конечном итоге найденное оптимальное решение больше нижней границы множества всевозможных маршрутов (этап 1), то оптимальность данного решения остается под вопросом.

Если существует оборванная ветвь с нижней границей меньшей, чем предварительно найденное решение, то предполагается возврат к оборванной ветви, поскольку, возможно, данная ветвь даст итоговое решение лучшее, чем предварительно найденное. Анализ оборванной ветви будет происходить аналогично алгоритму метода ветвей и границ без анализа внутренних оборванных ветвей.

Будем полагать, что оборванную ветвь в худшем случае мы пройдем с оценкой (15), при этом число рассматриваемых элементов матрицы (1) в худшем случае сократится на единицу (вследствие вычеркивания ребра, которое мы включали в предыдущие не оборванные ветви).

Если аналогичный возврат потребуется в дальнейшем, то число рассматриваемых элементов матрицы (1) вновь сократится на единицу.

В продолжение процесса может потребоваться, таким образом, не более n^2-4 возвратов. В результате оценка временной сложности худшего случая примет вид:

$$T\left(\frac{n^4}{6}-\frac{n^3}{4}\right)=O\left(n^7\log_2 n\right). \qquad (16)$$

Полученная оценка скрадывает эффект параллелизма за счет последовательного возврата к каждой оборванной ветви.

Можно заметить, однако, что каждую оборванную ветвь допустимо обрабатывать именно с момента обрыва на параллельной группе процессоров, динамически выделяемых этой ветви.

Каждой такой ветви потребуется не большее число процессоров, чем для исходной ветви, а в худшем случае число таких ветвей есть n^2-4 – по количеству обрабатываемых элементов матрицы, которое сокращается на единицу для каждой возвращаемой ветви.

Отсюда и из (16) требуемое число процессоров для параллельной обработки всех обрываемых ветвей составит $\left(n^2-4\right)\left(\frac{n^4}{6}-\frac{n^3}{4}\right)$. Каждая ветвь будет обрабатываться за время, не превосходящее правую часть (16), однако при этом начало обработки не детерминировано: оно может находиться в любом месте исходной ветви. Это означает, что общее время обработки всех ветвей не более чем удвоится.

Окончательно,

$$T\left(\frac{n^6}{6}\right)=O\left(n^5\log_2 n\right). \qquad (17)$$

Теорема 2. С учетом однократного возврата к каждой из отброшенных ветвей без вложений внутри них оценка временной сложности рассматриваемого алгоритма параллельного выполнения метода ветвей и границ примет вид (17).

Более сложный случай учета всех оборванных ветвей в данной работе не рассматривается.

Результаты эксперимента. Результаты эксперимента получены на основе численного моделирования изложенного выше параллельного алгоритма решения задачи коммивояжера методом ветвей и границ [13]. Параллельный алгоритм реализован последовательно на персональном компьютере. Результаты решения, получаемого по алгоритму вручную, всегда совпадают с программными результатами численного моделирования.

Пример 1. Входные данные заданы таблицей расстояний между городами (элементы матрицы a_{ij} соответствуют расстоянию от пункта i до пункта j):

i, j	**1**	**2**	**3**	**4**	**5**
1	∞	90	80	40	100
2	60	∞	40	50	70
3	50	30	∞	60	20
4	10	70	20	∞	50
5	20	40	50	20	∞

Результат решения вручную: (1,4), (4,3), (3,5), (5,2), (2,1). Длина маршрута равна $f = 180$.

Результат решения по программе: $(1,4), (4,3), (3,2), (2,5), (5,1)$. Длина маршрута равна $f = 180$.

Замечание 1. Можно заметить, что программа находит оптимальный путь, но при этом маршруты разные. Разница маршрутов появляется из-за того, что на этапе выбора дуги (ребра), по которой делается ветвление, находится несколько нулей, с одинаковыми рангами (степенями). При этом программа выбирает ближайший ноль в зависимости от предыдущей дуги (ребра), включенной программой в маршрут, а при расчете вручную на выбор дуги влияет субъективный фактор. Если подобные ситуации при решении не встречаются, то последовательность дуг (ребер), образующих маршрут, совпадает.

Проверка правильности работы алгоритма была проведена для различного количества городов и различных расстояний между ними, соответствующие примеры приведены в [13]. Результаты моделирования полностью подтверждают достоверность результатов работы предложенного алгоритма.

Сравнение с известными методами. Приведем таблицу сравнений временной сложности выполнения метода ветвей и границ, в последней строке приводится оценка предложенного алгоритма.

Авторы	Вид алгоритма и характер оценки	Оценка
Р.М. Колпаков, М.А. Посыпкин, И.Х. Сигал [14]	Нижняя оценка вычислительной сложности одной параллельной реализации метода ветвей и границ	$p \geq \frac{2 \cdot \sqrt{n(d-1)+2} - 3}{d-1}$, d – количество подзадач, n – общее число вершин.
Richard Wiener [15]	Верхняя оценка Последовательной реализации метода ветвей и границ	$O(n-1)!$
Leo Liberti [16]	Верхняя оценка Последовательной реализации метода ветвей и границ	$O(2^{2n^2} + (n-1)!)$
Авторы	Параллельная реализация метода ветвей и границ	$T\left(\frac{n^6}{6}\right) = O\left(n^5 \log_2 n\right)$

В [17, 18] алгоритмы и результаты эксперимента представлены на другой основе, без учета параллелизма отдельно взятой ветви. В [19, 20], а также в [3, 4] параллельные модификации метода ветвей и границ излагаются без аналитических оценок временной сложности. В [1, 2] и в [21] выполняется оптимизация последовательного решения рассматриваемой задачи.

Таким образом, предложенный метод отличается от известных по построению на основе максимально параллельной сортировки и тем, что он имеет полиномиальную верхнюю оценку временной сложности, в то время как последовательные методы характеризуются экспоненциальным порядком роста (факториальная оценка по формуле Стирлинга также имеет показательный порядок роста), выбранные для сравнения параллельные методы представлены без верхней оценки сложности отдельной ветви.

Заключение. Предложен алгоритм решения задачи коммивояжёра методом ветвей и границ, который максимально параллелен на каждом этапе нахождения нижней границы. Получены две оценки временной сложности полиномиального вида выполнения алгоритма для случаев без возвратов к оборванным ветвям и с возвратами к ним без учета вложений. При одновременной обработке всех обрываемых ветвей без учета вложений оценка временной сложности предложенного алгоритма сохраняет полиномиальный вид за счет роста числа процессоров. Данный алгоритм отличается от известных по построению и реализацией с полиномиальной временной сложностью на основе максимального распараллеливания каждой ветви метода. Численное моделирование предложенного алгоритма верифицирует правильность его работы.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Сергиенко И.В., Емец О.А., Емец А.О.* Задачи оптимизации с интервальной неопределенностью: метод ветвей и границ // Кибернетика и системный анализ. – 2013. – № 5. – С. 38-51.
2. *Овезгельдыев А.О., Морозов А.В.* Развитие метода ветвей и границ в задаче поиска оптимального кольцевого маршрута // Кибернетика и системный анализ. – 2013. – № 5. – С. 112-124.
3. *Gaurav Bhardwaj, Manish Pandey*. Parallel Implementation of the Max_Min Ant System for the Travelling Salesman Problem on GPU // International Journal of Computer Applications. – August 2014. –Vol. 99, No. 16. – P. 9-13.
4. *Gaurav Bhardwaj, Manish Pandey*. Parallel Implementation of Travelling Salesman Problem using Ant Colony Optimization // International Journal of Computer Applications Technology and Research. – 2014. – Vol. 3, Issue 6. – P. 385-389.
5. *Little J., Murty K., Sweeney D., Karel C.* An algorithm for the travelling salesman problem // Operations research. – 1963. – Vol. 11. – P. 972-989.
6. *Wirth N.* Algorithms and Data Structures. – August 2004. – P. 183.
7. *Ромм Я.Е., Назарьянц Е.Г.* Преобразование метода ветвей и границ для решения задачи коммивояжера на основе максимально параллельной сортировки. – Таганрог: ТГПИ., 2013. – 25 с. Деп. в ВИНИТИ 30.09.2013, № 279-В2013.
8. *Солодовников В.И.* Верхние оценки сложности решения систем линейных уравнений // В кн.: Теория сложности вычислений. I: Записки научных семинаров ЛОМИ АН СССР. – Л., 1982. – Т. 118. – С. 159-187.
9. *Гаврилкевич М.В., Солодовников В.И.* Эффективные алгоритмы решения задач линейной алгебры над полем из двух элементов // Обозрение прикладной и промышленной математики. – 1995. – Т. 2, № 3. – С. 399-439.
10. *Ромм Я.Е.* Параллельная сортировка слиянием по матрицам сравнений. I // Кибернетика и системный анализ. – 1994. – № 5. – С. 3-23.
11. *Ромм Я.Е.* Параллельная сортировка слиянием по матрицам сравнений. II // Кибернетика и системный анализ. – 1995. – № 4. – С. 13-37.
12. *Ромм Я.Е., Заика И.В.* Численная оптимизация на основе алгоритмов сортировки с приложением к дифференциальным и нелинейным уравнениям общего вида // Кибернетика и системный анализ. – 2011. – № 2. – С. 165-180.
13. *Ромм Я.Е., Дзюба А.С., Назарьянц Е.Г.* Преобразование и численное моделирование метода ветвей и границ для решения задачи коммивояжера на основе максимально параллельной сортировки. – Таганрог: ТГПИ, 2014. – 49 с. Деп. в ВИНИТИ 30.09.2014, № 279-В2013.
14. *Колпаков Р.М., Посыпкин М.А., Сигал И.Х.* О нижней оценке вычислительной сложности одной параллельной реализации метода ветвей и границ // Автоматика и телемеханика. – 2010. – № 10. – С. 156-166.
15. *Richard Wiener*. Branch and Bound Implementations for the Traveling Salesperson Problem // Journal of Object Technology. – March-April 2003. – Part. 1, № 2. – P. 65-86.
16. *Leo Liberti*. Branch-and-Bound for the Travelling Salesman Problem // LIX, Ecole Polytechnique, F-91128 Palaiseau. – March 15, 2011. – P. 1-8.
17. *Посыпкин М.А, Сигал И.Х.* Верхняя оценка для ускорения в одной параллельной реализации метода ветвей и границ решения задач дискретной оптимизации // Труды третьей международной конференции «параллельные вычисления и задачи управления», Москва 2-4 октября 2006. PACO'2006. – С. 897-908.

18. *Сигал И.Х., Бабинская Я.Л., Посыпкин М.А.* Параллельная реализация метода ветвей и границ в задаче коммивояжера на базе библиотеки BNB-Solver // Труды ИСА РАН. – 2006. – Т. 25. – С. 26-36.
19. *Посыпкин М.А, Сигал И.Х.* Применение параллельных эвристических алгоритмов для ускорения параллельного метода ветвей и границ // Журнал вычислительной математики и математической физики. – 2007. – Т. 47, № 9. – С. 1524-1537.
20. *Bazylevych R., Kuz B., Kutelmakh R.* Parallel Approaches for Solving Large-scale Travelling Salesman Problem // Second International Conference "Cluster Computing" CC. – June 3-5, 2013. – P. 30-34.
21. *Dimitrijevic V., Milosavljevic M., Markovic M.* Branch and bound algorithm for solving a generalized traveling salesman problem // Univ. Beograd. Publ. Elektrotehn. Fak. Ser. – Mat. 7 (1996). – P. 31-35.

REFERENCES

1. *Sergienko I.V., Emets O.A., Emets A.O.* Zadachi optimizatsii s interval'noy neopredelennost'yu: metod vetvey i granits [Задачи оптимизации с интервальной неопределенностью: метод ветвей и границ], *Kibernetika i sistemnyy analiz* [Cybernetics and Systems Analysis], 2013, No. 5, pp. 38-51.
2. *Ovezgel'dyev A.O., Morozov A.V.* Razvitie metoda vetvey i granits v zadache poiska optimal'nogo kol'tsevogo marshruta [Development of a method of branches and boundaries in the problem of finding the optimal ring route], *Kibernetika i sistemnyy analiz* [Cybernetics and Systems Analysis], 2013, No. 5, pp. 112-124.
3. *Gaurav Bhardwaj, Manish Pandey*. Parallel Implementation of the Max_Min Ant System for the Travelling Salesman Problem on GPU, *International Journal of Computer Applications*, August 2014, Vol. 99, No. 16, pp. 9-13.
4. *Gaurav Bhardwaj, Manish Pandey*. Parallel Implementation of Travelling Salesman Problem using Ant Colony Optimization, *International Journal of Computer Applications Technology and Research*, 2014, Vol. 3, Issue 6, pp. 385-389.
5. *Little J., Murty K., Sweeney D., Karel C.* An algorithm for the travelling salesman problem, *Operations research*, 1963, Vol. 11, pp. 972-989.
6. *Wirth N.* Algorithms and Data Structures, August 2004, pp. 183.
7. *Romm Ya.E., Nazar'yants E.G.* Preobrazovanie metoda vetvey i granits dlya resheniya zadachi kommivoyazhera na osnove maksimal'no parallel'noy sortirovki [The transformation method of branch and bound for solving the travelling salesman problem on the basis of maximum parallel sorting]. Taganrog: TGPI., 2013, 25 p. Dep. v VINITI 30.09.2013, No. 279-V2013.
8. *Solodovnikov V.I.* Verkhnie otsenki slozhnosti resheniya sistem lineynykh uravneniy [Upper bounds on the complexity of solving systems of linear equations], *V kn.: Teoriya slozhnosti vychisleniy. I: Zapiski nauchnykh seminarov LOMI AN SSSR* [In the book: The theory of computational complexity. I: notes of scientific seminars of LOMI an SSSR]. Leningrad, 1982, Vol. 118, pp. 159-187.
9. *Gavrilkevich M.V., Solodovnikov V.I.* Effektivnye algoritmy resheniya zadach lineynoy algebry nad polem iz dvukh elementov [Efficient algorithms for solving problems of linear algebra over the field of two elements], *Obozrenie prikladnoy i promyshlennoy matematiki* [Review of Applied and Industrial Mathematics], 1995, Vol. 2, No. 3, pp. 399-439.
10. *Romm Ya.E.* Parallel'naya sortirovka sliyaniem po matritsam sravneniy. I [Parallel merge sort on a matrix of comparisons. I], *Kibernetika i sistemnyy analiz* [Cybernetics and Systems Analysis], 1994, No. 5, pp. 3-23.
11. *Romm Ya.E.* Parallel'naya sortirovka sliyaniem po matritsam sravneniy. II [Parallel merge sort on a matrix of comparisons. I], *Kibernetika i sistemnyy analiz* [Cybernetics and Systems Analysis], 1994, No. 5, pp. 13-37.
12. *Romm Ya.E., Zaika I.V.* Chislennaya optimizatsiya na osnove algoritmov sortirovki s prilozheniem k differentsial'nym i nelineynym uravneniyam obshchego vida [Numerical optimization-based sorting algorithms with applications to differential and non-linear equations of the General form], *Kibernetika i sistemnyy analiz* [Cybernetics and Systems Analysis], 2011, No. 2, pp. 165-180.

13. *Romm Ya.E., Dzyuba A.S., Nazar'yants E.G.* Preobrazovanie i chislennoe modelirovanie metoda vetvey i granits dlya resheniya zadachi kommivoyazhera na osnove maksimal'no parallel'noy sortirovki [Transformation and numerical simulation method of branch and bound for solving the travelling salesman problem on the basis of maximum parallel sorting]. Taganrog: TGPI, 2014, 49 p. Dep. v VINITI 30.09.2014, No. 279-V2013.
14. *Kolpakov R.M., Posypkin M.A., Sigal I.Kh.* O nizhney otsenke vychislitel'noy slozhnosti odnoy parallel'noy realizatsii metoda vetvey i granits [On the lower bound computational complexity of a parallel implementation of the method of branches and borders], *Avtomatika i telemekhanika* [Automation and Remote Control], 2010, No. 10, pp. 156-166.
15. *Richard Wiener*. Branch and Bound Implementations for the Traveling Salesperson Problem, *Journal of Object Technology*, March-April 2003, Part. 1, No. 2, pp. 65-86.
16. *Leo Liberti*. Branch-and-Bound for the Travelling Salesman Problem, *LIX, Ecole Polytechnique, F-91128 Palaiseau,* March 15, 2011. – pp. 1-8.
17. *Posypkin M.A, Sigal I.Kh.* Verkhnyaya otsenka dlya uskoreniya v odnoy parallel'noy realizatsii metoda vetvey i granits resheniya zadach diskretnoy optimizatsii [An upper bound for the acceleration in a parallel implementation of the method of branch and bound for solving discrete optimization], *Trudy tret'ey mezhdunarodnoy konferentsii «parallel'nye vychisleniya i zadachi upravleniya», Moskva 2-4 oktyabrya 2006. RASO'2006* [Proceedings of the third international conference "parallel computations and control problems", Moscow, 2-4 October 2006. RASO'2006], pp. 897-908.
18. *Sigal I.Kh., Babinskaya Ya.L., Posypkin M.A.* Parallel'naya realizatsiya metoda vetvey i granits v zadache kommivoyazhera na baze biblioteki BNB-Solver [Parallel implementation of the method of branches and borders in the traveling salesman problem on the basis of the library BNB-Solver], *Trudy ISA RAN* [Proceedings of ISA RAS], 2006, Vol. 25, pp. 26-36.
19. *Posypkin M.A, Sigal I.Kh.* Primenenie parallel'nykh evristicheskikh algoritmov dlya uskoreniya parallel'nogo metoda vetvey i granits [Application of parallel heuristic algorithms for the parallel acceleration of the method of branches and borders], *Zhurnal vychislitel'noy matematiki i matematicheskoy fiziki* [Computational mathematics and mathematical physics], 2007, Vol. 47, No. 9, pp. 1524-1537.
20. *Bazylevych R., Kuz B., Kutelmakh R.* Parallel Approaches for Solving Large-scale Travelling Salesman Problem, *Second International Conference "Cluster Computing" CC*, June 3-5, 2013, pp. 30-34.
21. *Dimitrijevic V., Milosavljevic M., Markovic M.* Branch and bound algorithm for solving a generalized traveling salesman problem, *Univ. Beograd. Publ. Elektrotehn. Fak. Ser.* – Mat. 7 (1996), pp. 31-35.

Статью рекомендовал к опубликованию д.т.н. Г.Е. Веселов.

Ромм Яков Евсеевич – Таганрогский институт им. А.П. Чехова (филиал) Федерального государственного бюджетного образовательного учреждения высшего профессионального образования "РГЭУ (РИНХ)"; e-mail: romm@list.ru; 347926, г. Таганрог, ул. Инициативная, 48; тел.: 89094081126; д.т.н.; профессор; член Европейской Академии Естествознания (EuANH) зав. кафедрой информатики.

Назарьянц Елена Геворговна – e-mail: Elena2781@yandex.ru; тел.: 8515145004; аспирантка.

Romm Jacob Evseevich – Information Science of the State Educational Institution of Higher Professional Education of Taganrog Institute of Chekhov A.P. (branch) RGEU (RINH); e-mail: romm@list.ru; 48, Initsiativnaya street, Taganrog, 347926, Russia; phone: +79094081126; dr. of eng. sc.; professor; member of the European Academy of Natural Sciences (EuANH); chair of the department of Information Science.

Nazaryants Elena Gevorgovna – e-mail: Elena2781@yandex.ru; phone: +7515145004; postgraduate student.