Параллельная реализация метода ветвей и границ в задаче коммивояжера на базе библиотеки BNB-Solver*

И. Х. Сигал, Я. Л. Бабинская, М. А. Посыпкин

1. Введение

Впервые метод ветвей и границ был предложен Ленд и Дойг [1] в 1960 г. для решения общей задачи целочисленного линейного программирования. Интерес к этому методу и, фактически, его «второе рождение» связано с работой Литтла, Мурти, Суини и Кэрела, посвященной задаче коммивояжера [2]. Начиная с этого момента, появилось большое число работ, посвященных методу ветвей и границ и различным его модификациям. Столь большой успех объясняется тем, что авторы первыми обратили внимание на широту возможностей метода, отметили важность использования специфики задачи и сами воспользовались спецификой задачи коммивояжера.

Задача привлекает к себе внимание, так как в ней сочетаются простота постановки и трудность решения. Трудности имеют чисто вычислительный характер, ибо существование решения очевидно. В разные годы было предложено много методов решения задачи. Одни из них неэффективны, другие дают не обязательно оптимальное решение, и, наконец, некоторые требуют принятия интуитивных решений, что затрудняет программирование. Среди универсальных методов, гарантирующих оптимальное решение и удобных для программирования, наиболее эффективными, до появления метода ветвей и границ, оказались основанные на идеях динамического программирования. Эти методы дают экспоненциальную скорость роста времени решения задачи с ростом числа городов. С аналогичной скоростью растут требования к объему памяти. Большинство опубликованных методов для решения задач о коммивояжере подходят только для симметричных задач, в которых расстояние от города i до города j — то же, что от j до i, в то время как метод ветвей и границ годится и для асимметричных задач, появляющихся в различных приложениях.

Высокие требования к производительности процессоров и объемам памяти, которые предъявляет метод ветвей и границ, делают целесообразным применение многопроцессорных вычислительных комплексов (МВК). Создание параллельных программ, предназначенных для выполнения на таких системах, является сложной задачей, облегчить которую могут универсальные средства разработки, которыми являются, например библиотеки

* Работа выполнена при поддержке РФФИ, проекты 05–01–00495-а, 06–07–88079-а.

программ. В данной работе рассматривается опыт реализации параллельного варианта программы для решения задачи коммивояжера с помощью библиотеки BNB-Solver [5–7], которая позволяет свести к минимуму затраты, необходимые для создания параллельного приложения, основанного на методе ветвей и границ.

2. Постановка задачи

Задача о коммивояжере формулируется следующим образом. Пусть имеется n городов C_i (где $i = 0, 1, \ldots, n-1$) и задана матрица расстояний между ними A, где $a_{ij} \geqslant 0$ — расстояние между городами i и j.

$$A = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix}. \tag{1}$$

В случае симметричной задачи о коммивояжере $a_{ij} = a_{ji}$. В случае метрической задачи о коммивояжере $a_{ij} \leqslant a_{ik} + a_{kj}$, где $i, j, k \in \{0, 1, \ldots, n-1\}$.

Коммивояжер находится в городе C_0 и объезжает все города, побывав в каждом 1 раз, возвращается в город C_0.

Назовем маршрутом $z = (0, i_1, i_2, \ldots, i_{n-1}, 0)$, где $(i_1, i_2, \ldots, i_{n-1})$ — перестановка $(1, 2, \ldots, n-1)$. Будем считать, что $i_0 = i_n = 0$, тогда длина маршрута будет определяться следующим соотношением:

$$l(z) = \sum_{k=0}^{n-1} a_{i_k i_{k+1}}. \tag{2}$$

P — множество всех таких маршрутов. Требуется найти $z_0 \in P$:

$$l(z_0) = \min_{z \in P} l(z). \tag{3}$$

Множество P (допустимых решений) в этой задаче — это множество n-мерных переменных $z = (i_0, i_1, \ldots, i_n)$ с компонентами $0, 1, \ldots, n$, удовлетворяющих условиям:

$$\begin{gathered} i_0 = i_n = 0, \\ (i_1, i_2, \ldots, i_{n-1}) \text{ — перестановка } \quad (1, 2, \ldots, n-1). \end{gathered} \tag{4}$$

Заметим, что $|P| = (n-1)!$.

3. Алгоритм метода ветвей и границ, применительно к решению задачи коммивояжера

Под методом ветвей и границ понимается алгоритм решения задачи, имеющей древовидную структуру поиска оптимального решения и использующий результаты решения оценочных задач. Древовидная структура

называется обычно деревом ветвления. Подробно общая схема алгоритма ветвей и границ рассмотрена в [3, 4], а для задачи коммивояжера в [2, 4]. Алгоритм метода ветвей и границ применительно к задаче коммивояжера заключается в следующем:

1. Множество путей разбивается на 2 подмножества, содержащее непосредственный переход из вершины i в вершину j и не содержащее его. Для каждого из классов находится нижняя оценка.
2. Разбиение одного из уже существующих подмножеств на два подмножества по выше описанному методу. Нахождение для них нижних оценок. Если одно из них является уже построенным путем, сравнение длины этого пути с рекордом (наилучшим найденным к текущему моменту значением) и изменение рекорда, если необходимо.
3. Отсев.
4. Повторение 2-го и 3-го шагов, пока останется хотя бы одно не отсеянное подмножество вершин.

Основная трудность этого алгоритма заключается в поиске и запрещении циклов размерностью меньшей, чем размерность задачи.

4. Описание параллельной реализации метода ветвей и границ

4.1. Общая схема параллельного алгоритма

Основная сложность распараллеливания метода ветвей и границ заключается в том, что он ведет себя непредсказуемо, т. е. структура дерева ветвления заранее не известна и строится в процессе решения. Эта структура, по существу, является динамической и это существенно затрудняет распараллеливание. Некоторые сведения по этому поводу содержатся в [5, 6].

Существует несколько подходов к распараллеливанию метода ветвей и границ, в данной работе мы следовали следующему подходу. Выделенный процессор, называемый *мастер-процессором*, управляет равномерным распределением задач по остальным процессорам, называемых *рабочими*. При этом он может как производить ветвления так и управлять остальными процессорами. Каждый рабочий процессор получает подзадачу или набор подзадач. После выполнения определенного количества ветвлений или получения определенного количества подзадач, процессор посылает сообщение мастеру, что предел достигнут. Мастер находит свободный процессор и пересылает ему часть подзадач. В случае если свободного процессора нет, задачи хранятся на мастере до тех пор, пока один из процессоров не сообщит об окончании решения всех своих подзадач.

4.2. Интеграция с библиотекой BNB-Solver

Программа была написана на языке C++ с использованием библиотеки BNB-Solver [5–7]. Библиотека BNB-Solver предоставляет поддержку для параллельной реализации метода ветвей и границ. Для интеграции новой задачи в библиотеку требуется реализовать проблемно-зависимый

класс, предоставляющий функции для операции ветвления, вычисления оценки, и еще ряд методов и описаний. Этот класс используется в качестве шаблонного параметра при создании экземпляров объектов последовательного и параллельного решателей. Например, для создания параллельного решателя используется следующая конструкция:

```
BNBClmsSolver <TSPFactory, WTraverse <TSPFactory>> solver(...);
```

Классы `BNBClmsSolver` и `WTraverse`, реализующие описанную выше стратегию распараллеливания и схему фронтального ветвления соответственно, предоставляются библиотекой, а класс `TSPFactory`, содержащий детали, специфичные для задачи коммивояжера, реализуется пользователем библиотеки. Далее для решения задачи требуется вызвать метод `solve` созданного класса `solver`, после чего можно воспользоваться методами для доступа к результатам расчетов.

Для решения задачи коммивояжера было написано две программы для последовательного и параллельного варианта соответственно. Результаты экспериментов, приведенные в следующем разделе, были получены при помощи параллельного варианта программы.

5. Результаты экспериментов

5.1. Описание вычислительных экспериментов

Вычислительный эксперимент заключался в получении зависимости времени решения задачи от количества процессоров и от размерности задачи. Поскольку, в случае задачи коммивояжера время решения очень сильно зависит от начальных данных, была создана серия из 20 задач размерностями от 21 до 40 городов, где каждая задача отличалась от предыдущей добавлением одного города, при этом матрица расстояний между остальными городами оставалась неизменной. Эта серия задач была создана из тестовой задачи размерности 42, опубликованной в [10]. Это позволило свести к минимуму зависимость времени решения задачи от самой матрицы расстояний и получить зависимость времени решения от размерности.

Эта серия тестовых задач была решена для разного количества процессоров на суперкомпьютере MVS-15000 BM, установленном в Межведомственном суперкомпьютерном центре РАН [8]. В состав этого комплекса входят 1048 процессоров PowerPC 2.2 GHz.

5.2. Результаты и анализ результатов

Ниже приводятся графики, иллюстрирующие зависимость времени решения от количества процессоров (рис. 1, 2).

Как видно из этих графиков при увеличении количества процессоров эффективность работы растет достаточно быстро. Кроме того, видно, что вид графиков зависимости времени решения от количества процессоров для всей серии задач практически идентичен. Так же можно заметить, что при увеличении количества процессоров с 1-го до 2-х ускорения не происходит. Причиной этого является то, что при наличии одного рабочего

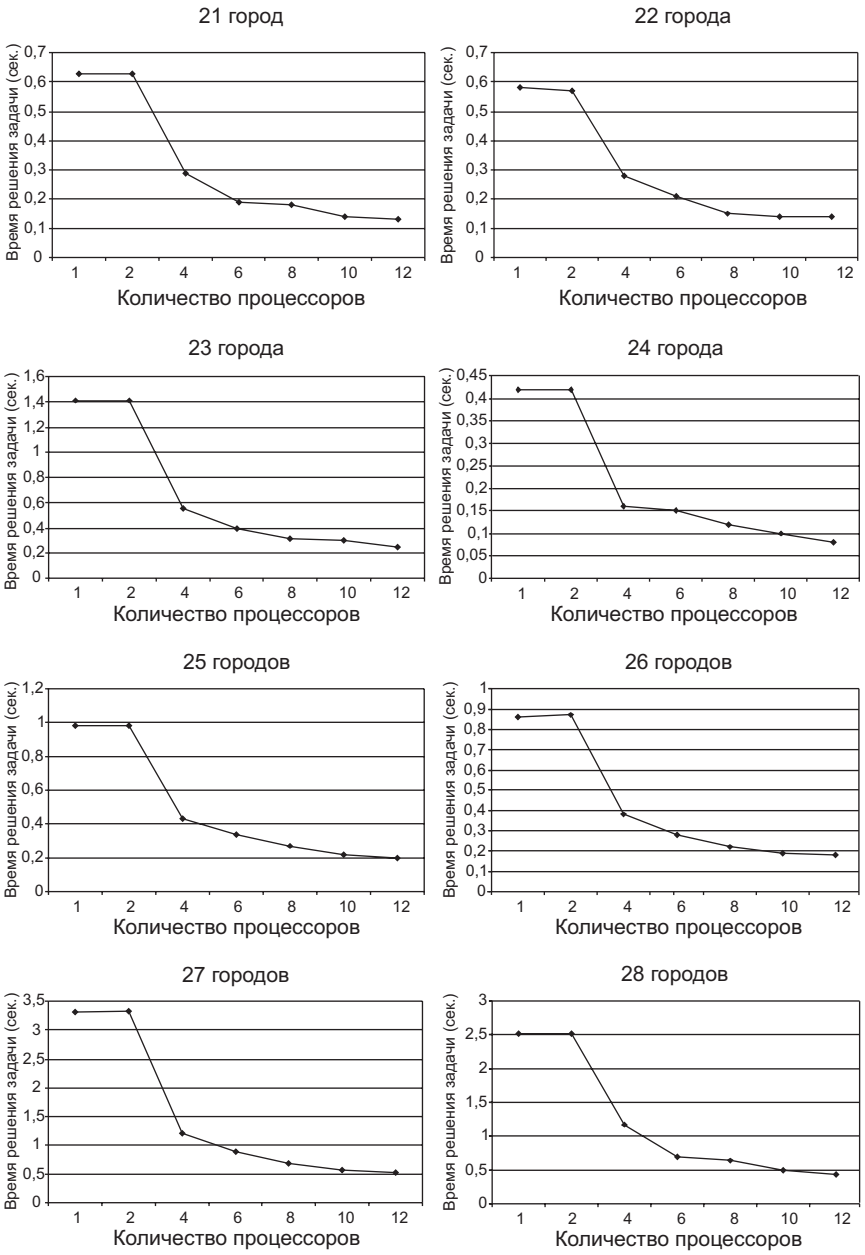

Рис. 1. Зависимость времени решения задачи от количества процессоров для серии тестовых задач

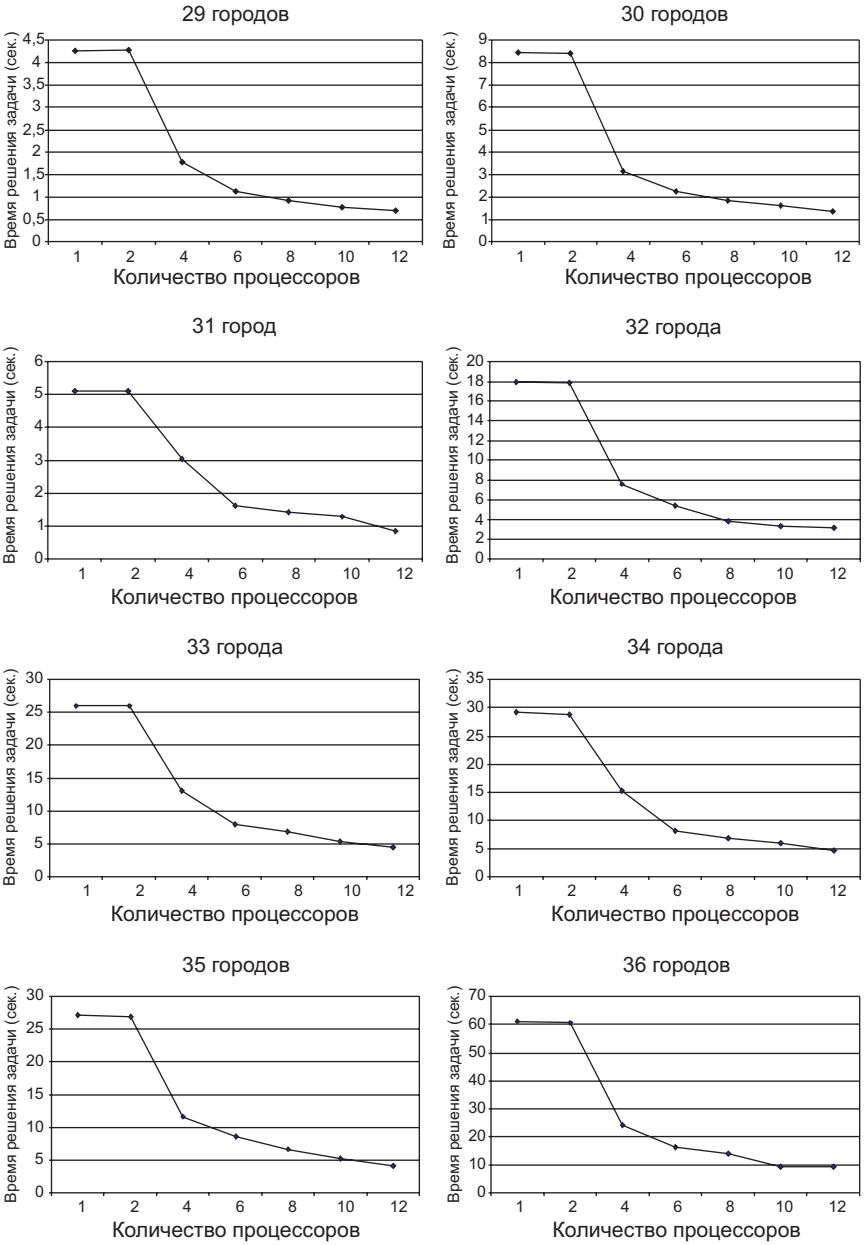

Рис. 1 (продолжение)

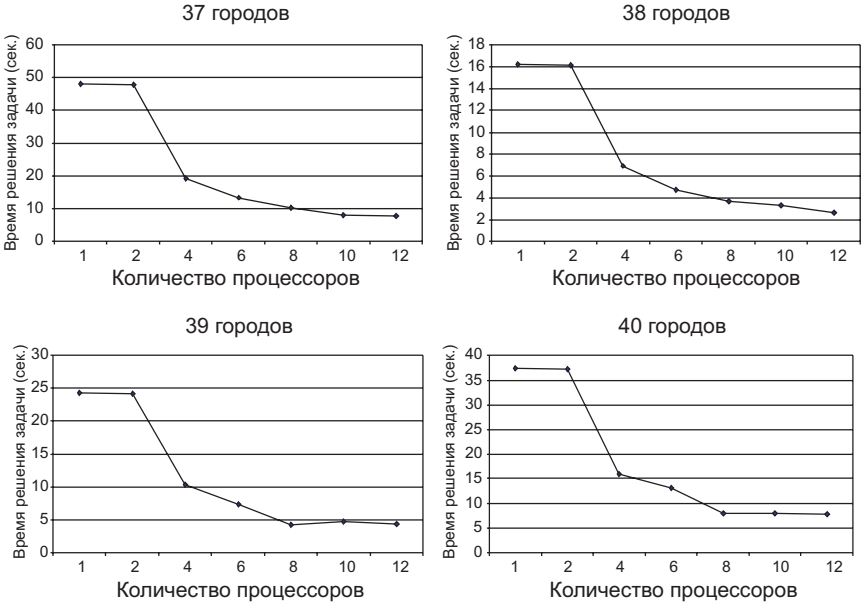

Рис. 1 (окончание)

Зависимость времени решения от количества процессоров для серии тестовых задач

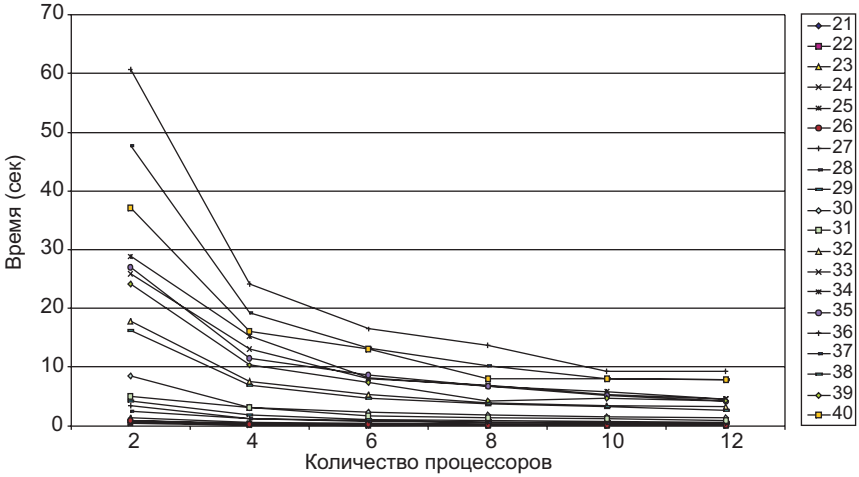

Рис. 2. Объединенный график для всей серии задач

Рис. 3. Полученное максимальное (для числа процессоров от 1 до 12) ускорение в серии задач размерностями 21–40

Рис. 4. Зависимость времени решения задачи от ее размерности

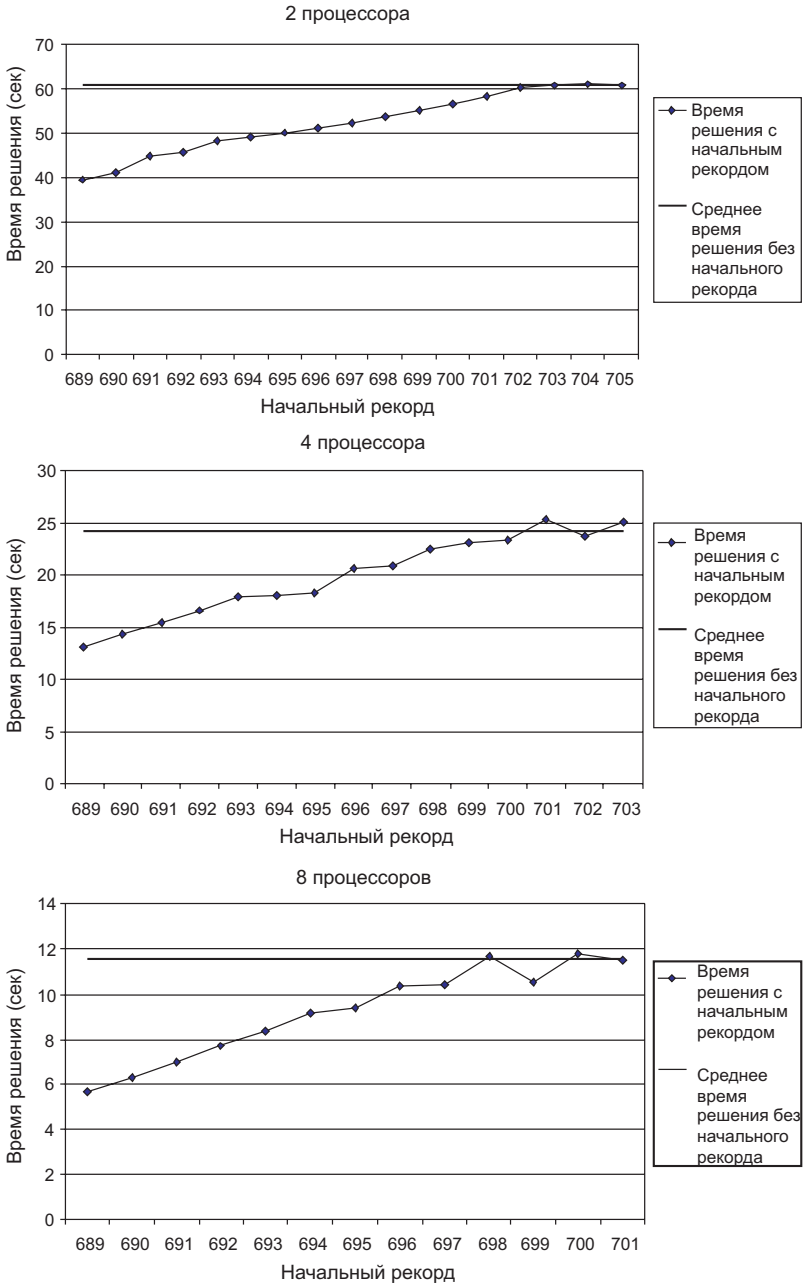
2 процессора
Время решения (сек)
70
60
50
40
30
20
10
0
689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705
Начальный рекорд
Время решения с начальным рекордом
Среднее время решения без начального рекорда
4 процессора
Время решения (сек)
30
25
20
15
10
5
0
689 690 691 692 693 694 695 696 697 698 699 700 701 702 703
Начальный рекорд
Время решения с начальным рекордом
Среднее время решения без начального рекорда
8 процессоров
Время решения (сек)
14
12
10
8
6
4
2
0
689 690 691 692 693 694 695 696 697 698 699 700 701
Начальный рекорд
Время решения с начальным рекордом
Среднее время решения без начального рекорда

Рис. 5

процесса управляющий процесс сам не производит ветвлений, а только передает задания на счет рабочему процессу.

Ниже приводится диаграмма, отражающая максимальное полученное ускорение (рис. 3) и график зависимости времени решения задачи от ее размерности (рис. 4).

Как видно из этого графика время решения задачи при увеличении размерности задачи очень быстро растет, но продолжает сильно зависеть от начальных данных, несмотря на то, что различие начальных данных в этой серии задач было сведено к минимуму.

Для получения зависимости времени решения задачи от близости начального рекорда к решению, была выбрана наихудшая (по времени решения) задача из серии. Это задача из 36 городов. Графики, отражающие данную зависимость, приводятся на рис. 5.

Как видно из этих графиков, при увеличении числа процессоров требуется более близкий к решению начальный рекорд для того, чтобы был выигрыш во времени. Это происходит из-за того, что процессоры не обмениваются рекордами в реальном времени, а обновляют рекорд лишь при получении очередного набора вершин для ветвления.

6. Заключение

Реализованы параллельные и последовательные алгоритмы решения задачи о коммивояжере на основе метода типа ветвей и границ и различные алгоритмы нахождения начального рекорда. Исходя из полученных результатов, можно сделать следующие выводы. Выяснилось, что время решения задачи довольно быстро увеличивается с ростом размерности задачи, но очень сильно зависит от начальных данных. Именно по этой причине тестовая задача из 36 городов решается дольше, чем тестовые задачи размерностей 37, 38, 39 и 40, полученные из нее добавлением 1-го, 2-х, 3-х и 4-х городов соответственно. Эксперимент показал, что близкий к оптимуму начальный рекорд может существенно сократить время решения задачи, причем, чем больше количество процессоров, тем ближе должен быть начальный рекорд к решению, для получения выигрыша во времени.

Следует отметить, что разработка алгоритмов, описанных в данной работе, находится в начальной стадии. Например, в результате того, что для каждой вершины приходится хранить матрицу расстояний целиком, машинных ресурсов не хватает, для того чтобы решать задачи размерности больше 40. Так же представляется интересным понять, как определить, сколько процессоров нужно для рационального решения данной задачи. Для решения этих и многих других вопросов предлагается разработка математической модели процесса параллельных вычислений, обширный вычислительный эксперимент и совместный анализ полученных результатов.

Программы были написаны с использованием библиотеки BNB-Solver. Ранее эта библиотека применялась для решения задачи о ранце с одним и несколькими ограничениями [5, 6]. В данной работе описано применение того же программного средства для решения задачи коммивояжера,

которая существенно отличается от задачи о ранце. Это показывает, что с применением библиотеки BNB-Solver можно решать различные задачи в рамках метода ветвей и границ.

Литература

1. *Land A. H.*, *Doig A. G.* An autmatic method of solving discrete programming problems // Econometrica. Vol. 28 (1960). P. 497–520.
2. *Литтл Дж.*, *Мурти К.*, *Суини Д.*, *Кэрел К.* Алгоритм для решения задачи о коммивояжере // Экономика и математические методы. Т. 1. Вып. 1 (1965). С. 94–107.
   Little J. D. C., *Murty K. G.*, *Sweeney D. W.*, *Karel C.* An algorithm for the traveling salesman problem. Operations Research. Vol. 11 (1963). P. 972–989.
3. *Корбут А. А.*, *Финкельштейн Ю. Ю.* Дискретное программирование. М.: Наука, 1969.
4. *Сигал И. Х.*, *Иванова А. П.* Введение в прикладное дискретное программирование: модели и вычислительные алгоритмы: Учеб. пособие. М.: ФИЗМАТЛИТ, 2002.
5. *Посыпкин М. А.*, *Сигал И. Х.* Исследование алгоритмов параллельных вычислений в задачах дискретной оптимизации ранцевого типа // Журнал вычислительной математики и физики. 2005. № 10. Т. 45. С. 1801–1809.
6. *Посыпкин М. А.*, *Сигал И. Х.*, *Галимьянова Н. Н.* Алгоритмы параллельных вычислений для решения некоторых классов задач дискретной оптимизации. Сообщения по прикладной математике. М.: ВЦ РАН, 2005.
7. *Посыпкин М. А.* Архитектура и программная организация библиотеки для решения задач оптимизации методом ветвей и границ на многопроцессорных вычислительных комплексах. См. статью в этом сборнике.
8. Ссылка в Интернете: www.jscc.ru.
9. *Karp R. M.* Management science. Heuristic approach to solving salesman problems. № 10 (1964). P. 225–248.
10. *Groes G. A.* A method for solving traveling salesman problem. Operations Research. Vol. 6 (1958). P. 795.