Решение задачи коммивояжера на многопроцессорных системах с общей и распределенной памятью *

А. Л. Игнатьев[1], М. А. Посыпкин[2], И. Х. Сигал[1]

[1] *Вычислительный центр РАН*
[2] *Центр Грид-технологий и распределенных вычислений Института системного анализа РАН*

Задача коммивояжера заключается в построении минимального маршрута объезда нескольких городов с заданными попарными расстояниями. Рассматривается численное решение этой задачи на многопроцессорных системах с общей и распределенной памятью. Приводится описание различных подходов к реализации точных и приближенных методов решения этой задачи, результатов вычислительного эксперимента.

В статье рассматривается известная задача коммивояжера, которая формулируется следующим образом. Пусть имеется $n + 1$ город с номерами 0, 1, …, n; для каждой пары городов i и j задано «расстояние» $c_{ij} \geq 0$ между ними, таким образом задается матрица «расстояний». Коммивояжер должен, выехав из начального города 0, объехать все города, посетив каждый из них ровно один раз, и вернуться в город с номером 0. Необходимо найти такую последовательность посещения городов, при которой суммарная длина пути минимальна [1]. Асимметричной называется такая задача коммивояжера, где условие $c_{ij} = c_{ji}$ в общем случае не выполняется.

Многопроцессорные вычислительные комплексы можно условно разделить на две группы: системы с общей и распределенной памятью. Параллельные системы с общей памятью характеризуются тем, что все вычислительные устройства (процессоры, ядра) разделяют общее адресное пространство. Это упрощает написание приложений для них, так как не надо заботиться о копировании данных между процессорами. С другой стороны, появляется проблема синхронизации общего доступа к данным.

* Работа выполнена при поддержке РФФИ (№ 08–07–00072–а) и Совета по грантам Президента Российской Федерации (№ НШ–5511.2008.9).

Кроме того, такие системы плохо масштабируются. Многопроцессорные системы с распределенной памятью представляют собой набор вычислительных модулей, каждый из которых состоит из процессора и оперативной памяти, соединенных между собой высокоскоростной сетью. Такие системы гораздо лучше масштабируются, но написание приложений для них более трудоемко.

Методы решения задачи коммивояжера [2] можно условно разделить на две категории: точные или ε-приближенные, содержащие доказательства оптимальности или ε-оптимальности и приближенные, не содержащие таких доказательств. К первой категории относятся методы последовательного анализа вариантов, ветвей и границ, динамического программирования и другие. Основным недостатком методов нахождения точного (или ε-приближенного) решения является их высокая вычислительная сложность.

Методы, не содержащие доказательства оптимальности, часто называются приближенными. В приближенных методах решение задачи производится в два этапа: построение начального решения и его улучшение. На первом этапе широко используются эвристические алгоритмы, опирающиеся на правдоподобные, но строго не обоснованные предположения о свойствах оптимального решения задачи. На втором этапе применяются алгоритмы локальной оптимизации, связанные с введением понятия окрестности и изменением этой окрестности и последовательности работы самих алгоритмов в процессе решения задачи.

Метод ветвей и границ носит переборный характер и основан на идее разбиения множества допустимых решений задачи на подмножества (ветвление), постепенно исключающиеся из рассмотрения с помощью процедуры отсева с использованием значений нижней границы (ее оценки). Процедура оценки нижней границы позволяет определить, содержится ли в рассматриваемом подмножестве решение, заведомо лучшее, чем уже найденное ранее (рекорд) в соответствии с правилами отсева в методе ветвей и границ [1]. Подмножества исключается из дальнейшего рассмотрения (отсеиваются), когда нижняя граница на подмножестве превосходит либо совпадает с рекордом. Если в процессе вычислений находится решение, лучше известного к очередному шагу алгоритма, оно становится новым рекордом. Эффективная процедура оценки нижней границы позволяет сократить число рассматриваемых подмножеств за счет их более быстрого отсева и в ряде случаев приводит к значительному увеличению производительности. Метод ветвей и границ имеет декомпозиционную структуру и поэтому обладает высоким потенциалом распараллеливания: после ветвления множества решений каждое полученное подмножество может обрабатываться независимо от остальных.

Известно, что задача коммивояжера принадлежит к классу NP-сложных задач и поэтому требует больших вычислительных ресурсов таких,

что мощностей обычных рабочих станций часто оказывается недостаточно. В связи с этим для ее решения представляется целесообразным применение методов параллельных вычислений. Предлагаемый алгоритм был реализован с помощью библиотеки BNB-Solver [3], предназначенной для решения задач дискретной и глобальной оптимизации методом ветвей и границ на параллельных вычислительных системах с общей и распределенной памятью. Данная библиотека имеет гибкую модульную программную инфраструктуру, основанную на шаблонах C^{++}, обеспечивающую удобное подключение новых задач оптимизации. При этом требуется реализовать только проблемно-зависимые компоненты.

В случае систем с общей памятью применяется схема с одним общим списком подзадач и локальными для каждого потока списками. Каждый поток выполняет операции метода ветвей и границ, используя свой список подзадач. При этом периодически подзадачи добавляются в общий список и извлекаются из него потоками в случае, когда локальный список становится пустым.

Общая идея реализации МВГ на системах с распределенной памятью заключается в том, что каждый процессор работает со своим списком подмножеств, сохраняемым в локальной памяти. Если в результате выполнения разбиений локальный список становится пустым, то для предотвращения простоев дополнительные множества пересылаются с другого процессора. Задача считается решенной, когда на всех процессорах списки не содержат подмножеств. Применяются централизованные схемы организации вычислений: выделенный управляющий процессор отправляет рабочим процессорам области поиска и последний рекорд. Эти процессоры выполняют разбиения полученного подмножества, при необходимости пересылая на управляющий процессор часть подмножеств. Более подробно реализация методов балансировки для систем с общей и распределенной памятью рассмотрена в работе [4].

В качестве системы с распределенной памятью использовался вычислительный кластер МВС100К [5], оснащенный 1980-ю четырехъядерными процессорами Intel Xeon 5365 3 ГГц. В качестве системы с общей памятью была выбрана рабочая станция с четырехъядерным процессором Intel Core 2 Quad 9550 2.83 ГГц, 4 Гб оперативной памяти. Далее в случае распределенной памяти используется понятие «процессор» для обозначения одного вычислительного устройства. В случае суперкомпьютера МВС100К, использующего многоядерные процессоры, под процессором будет пониматься одно ядро. Такая терминология вполне допустима и даже желательна, т. к. в случае с распределенной памятью ядро рассматривается как отдельное вычислительное устройство.

Целью первой серии экспериментов было выявление лучшей оценочной процедуры для вычисления значений нижней границы их двух: первой,

основанной на алгоритме приведения матриц расстояний [6], а второй — на решении задачи о назначении [1]. Задача о назначении формулируется следующим образом. Пусть имеется n работ и n кандидатов, причем назначение кандидата i на работу j требует затрат $c_{ij} \geq 0$. Требуется так распределить кандидатов по работам, чтобы (*i*) суммарные затраты были минимальными и (*ii*) каждый кандидат может быть назначен только на одну работу и на каждую работу назначается только один кандидат [1].

Предположим, что матрица $X = (x_{ij})$, такая, что $x_{ij} = 1$, если работник i назначен на работу j, и $x_{ij} = 0$ в противном случае, является решением задачи о назначении. Из условия (*ii*) следует, что в каждой строке и в каждом столбце матрицы находится только одна единица. Поставим в соответствие матрице X n-вершинный ориентированный граф. В этом графе существует дуга (i, j), если $x_{ij} = 1$. Получим один или несколько циклов. Случай с одним циклом соответствует задаче коммивояжера. Действительно, задача о назначении является задачей коммивояжера с отсутствующим условием существования только одного цикла (условия посещения каждого города ровно один раз и возврата в начальный город выполняются только в случае одного цикла).

В работе произведено сравнение двух алгоритмов вычисления оценки для различных вариантов задачи коммивояжера из библиотеки TSPLIB [7]. Полученные результаты, представленные в табл. 1, показывают, что процедура оценки, основанная на задаче о назначениях, является более предпочтительной. Поэтому эта процедура выбрана для проведения последующих экспериментов. Результаты экспериментов для процедур оценки, основанной на приведении матриц, содержатся в [9].

Результаты эксперимента для задачи Ftv90 (91 город) представлены в табл. 2. В первом столбце содержится число использованных процессоров, во втором — время решения задачи T, в третьем — вычислительная сложность S, выражающаяся в числе ветвлений, и в четвертом — производительность $P = S/T$.

Результаты, приведенные в табл. 2, показывают, что время решения определяется не только числом задействованных процессоров, но и изменением количества ветвлений. Процесс решения задачи методом ветвей и

Таблица 1

Время решения задач с использованием двух процедур оценки нижней границы на 64 процессорах (сек)

Задача	Ftv33	Ftv38	Ftv47	Ftv55
Приведение матриц	0,12	0,21	3,61	108,02
Задача о назначении	0,11	0,08	1,64	5,03

Таблица 2

Результаты эксперимента для распределенной памяти для задачи Ftv90

Число процессоров	**Время (сек), *T***	**Число ветвлений, *S***	**Производительность, *P***
1	664.83	3299763	4963,30
2	96,90	1331230	13738,65
4	34,49	900641	26116,81
8	42,03	1939826	46153,93
16	26,14	2374063	90818,68
32	21,81	3225846	147886,08
48	8,52	2038415	239296,00
64	4,11	1325692	322588,43
96	2,9	1194724	411675,26

границ можно представить как обход дерева, в каждой вершине которого по соотношению между оценкой и рекордом, определяется — подлежит ли эта вершина дальнейшему ветвлению. Зависимость количества ветвлений от числа процессоров объясняется тем, что при изменении порядка обработки вершин дерева меняется темп улучшения рекорда, что в свою очередь, влияет на интенсивность отсева и общее число обработанных вершин. Вследствие этого время решения может вести себя нелинейно и даже немонотонно при увеличении числа процессоров (4 и 8 процессоров в табл. 2). Поэтому в таблице приводятся значения еще одного параметра — производительности, отражающего скорость обработки вершин дерева ветвления. Производительность увеличивается практически пропорционально числу задействованных процессоров, что подтверждает высокую масштабируемость алгоритма.

Чтобы нивелировать влияние порядка обхода, была проведена еще одна серия экспериментов, в которой перед началом ветвлений рекорду присваивается оптимальное значение. Тогда порядок обхода дерева становится несущественным, так как оптимальное решение уже известно, и производится лишь отсев вершин (т. е. доказательство оптимальности). Полученные результаты (см. табл. 3) показывают, что число вершин в дереве ветвлений практически не изменяется и наблюдается снижение времени решения при увеличении числа процессоров с 1 до 64. Уменьшение производительности в случае с 96 процессорами объясняется увеличивающимися потерями в связи с межпроцессорными обменами данными, влияние которых становится особенно ощутимым при малом общем времени решения задачи.

Таблица 3

Результаты эксперимента (задача Ftv90) для распределенной памяти с заданным рекордом, равным оптимальному значению

Число процессоров	Время (сек), T	Число ветвлений, S	Производительность, *P*
1	56,53	432703	7654,54
2	28,31	432701	15284,58
4	14,14	432697	30608,72
8	7,83	432689	55268,24
16	4,59	432673	94342,87
32	2,64	432641	164097,71
48	1,72	432609	251570,98
64	1,6	432577	270186,19
96	1,7	432513	254532,90

По результатам двух предыдущих экспериментов видно, что быстрое получение близкого к оптимуму рекорда (в нашем случае равного оптимуму) многократно ускоряет решение задачи. Одним из способов улучшения значения рекорда является применение алгоритмов локальной оптимизации (эвристический поиск). Был реализован алгоритм локальной оптимизации, на каждом шаге которого происходит перестановка двух соседних городов в маршруте. При этом все доступные процессоры делятся на две группы: занятые ветвлением, и эвристическими процедурами. Если удается получить новое улучшенное значение рекорда, то он рассылается всем процессорам, занятым ветвлением, что ускоряет отсев подмножеств. Результаты работы такого алгоритма представлены в табл. 4, откуда видно, что увеличение числа процессоров, отведенных под эвристический поиск, уменьшает число ветвлений. В то же время уменьшается число процессоров, занятых ветвлением. В рассмотренном примере наилучшим является распределение процессоров поровну (по 32) между ветвлением и эвристическим поиском. Ускорение за счет применения эвристики в этом случае составляет почти 30 %.

Результаты эксперимента для систем с общей памятью представлены в табл. 5. Реализация с общей памятью показывает меньшую масштабируемость, чем реализация с распределенной памятью. Этот факт является следствием потерь на синхронизацию потоков при использовании общих ресурсов.

В табл. 6. приведено время решения различных задач с распределенной памятью на 64 процессорах. В восьми примерах, взятых из [3], число городов колеблется от 45 до 444.

Таблица 4

Результаты использования распределения 64 процессоров по двум группам (задача Ftv100)

Число процессоров, реализующих эвристики	**Время (сек), *T***	**Число ветвлений, S**	**Производительность, *P***
0	51,35	8035502	156475,48
8	58,08	7833920	134892,55
16	50,72	6702488	132136,56
32	36,28	5286109	145705,04
48	46,45	3994667	85996,70

Таблица 5

Результаты эксперимента для общей памяти для задачи Ftv47

Число ядер	1	2	3	4
Время (сек)	7,07	3,98	2,67	2,23

Таблица 6

Время решения различных задач

Задача	**Число точек**	**Время (сек)**
Ftv44	45	0.09
Ftv90	91	4.1
Ftv100	101	51.35
Ftv110	111	888
Rbg323	324	277
Rbg358	359	553
Rbg403	404	1259
Rbg443	444	2523

Сформулируем основные выводы по приведенным результатам экспериментальных исследований:

- время решения задач с применением процедуры назначений существенно меньше, чем с использований процедуры приведений; это превосходство тем больше, чем больше размерность решаемых задач;
- время решения задач с применением распределенной памяти при увеличении числа процессоров имеет тенденцию к монотонному уменьшению, а производительность (отношение числа ветвлений ко времени) — к монотонному росту;

- использование эвристических процедур для построения рекордов в процессе решения существенно сокращает время вычислений; число ветвлений монотонно уменьшается при росте числа процессоров.

Литература

1. *Сигал И. Х., Иванова А. П.* Введение в прикладное дискретное программирование: модели и вычислительные алгоритмы: Учеб. пособ., 2-е изд., испр. и доп. М.: Физматлит, 2007.
2. *Игнатьев А. Л.* Сравнение различных методов решения задачи коммивояжера на многопроцессорных вычислительных машинах // Современные информационные технологии и ИТ-образование: III Межд. науч.-практ. конф., Москва, МГУ им. М. В. Ломоносова, 18–21 декабря 2006 г.: Сб. трудов под ред. В. А. Сухомлина, М.: МАКС Пресс, 2008. С. 509–513.
3. *Посыпкин М. А.* Архитектура и программная реализация библиотеки для решения задач оптимизации методом ветвей и границ на многопроцессорных вычислительных комплексах // Проблемы вычислений в распределенной среде: распределенные приложения, коммуникационные системы, математические методы и оптимизация. ИСА РАН. М.: КомКнига/URSS, 2006. С. 18–25.
4. *Evtushenko Y., Posypkin M., Sigal I.,* A framework for parallel large-scale global optimization // Computer Science — Research and Development 23(3), P. 211–215, 2009.
5. Вычислительный кластер МВС100К: http://www.jscc.ru/hard/mvs100k.shtml
6. *Литл. Дж., Мурти К., Суини Л., Кэрел К.* Алгоритм для решения задачи коммивояжера // Экономика и математические методы. 1965. Т. 1. В. 1. С. 94–107.
7. *Reinelt G.* TSPLIB — A Traveling Salesman Problem Library // ORSA Journal on Computing. 1991. № 3. P. 376–384.
8. *Игнатьев А. Л.* Параллельные методы решения задачи коммивояжера // Труды 51-й научной конференции МФТИ «Современные проблемы фундаментальных и прикладных наук»: Часть IX. Инновации и высокие технологии. М.: МФТИ, 2008. С. 4–6.