РОССИЙСКАЯ АКАДЕМИЯ НАУК

СИБИРСКОЕ ОТДЕЛЕНИЕ

АВТОМЕТРИЯ

№ 6 2001

УДК 519.854.3

Г. И. Забиняко

(Новосибирск)

ПАРАЛЛЕЛЬНЫЙ АЛГОРИТМ ЦЕЛОЧИСЛЕННОЙ ЛИНЕЙНОЙ ОПТИМИЗАЦИИ*

Рассматривается параллельный алгоритм ветвей и границ для задач целочисленной линейной оптимизации и его реализация на многопроцессорном компьютере с общей памятью RM600 на Фортране 77 с использованием системы параллельного программирования MPI. Производится сопоставление эффективности параллельного и последовательного алгоритмов.

Введение. Алгоритм предназначен для решения задач следующего вида: найти максимум функции $f(x)=(c,x)$ при ограничениях

$$Ax=b,\quad \alpha \le x \le \beta,\ \text{ компонента } x_j \text{ – целое},\quad j\in J.$$

Здесь A – матрица размера $m\times n$; J – список номеров целочисленных переменных; $c, x, \alpha, \beta \in R^n$, $b\in R^m$.

Нецелочисленные переменные x_j, $j\notin J$, могут быть ограниченными только снизу или сверху либо свободными. Можно фиксировать любые из переменных, полагая $\alpha_j=\beta_j$, и общие ограничения задавать в виде неравенств.

Параллельная версия алгоритма используется в пакете программ на этапе численного решения задач. Входные данные пакета представляются в общепринятом для задач математического программирования MPS-формате [1]. В этом формате накоплены большие наборы тестовых задач, относящиеся к различным областям применения: оптимизации технических систем, проектированию устройств и др.

Распараллеливание осуществляется исполнением на каждом из процессоров алгоритма ветвей и границ с односторонним ветвлением [2].

1. **Алгоритм ветвей и границ с односторонним ветвлением.** Алгоритм основывается на решении серии оценочных задач линейного программирования (ЛП). Выбор переменной ветвления производится с помощью штрафов [1, 2]. Базисную переменную x_j для $j\in J$ в оптимальном базисе очередной оценочной задачи ЛП представим в виде $x_j=[x_j]+v_j$, где $[x_j]$ – целая

* Работа выполнена при поддержке Российского фонда фундаментальных исследований (проект № 99-07-90422).

часть x_j. Штраф за увеличение x_j на величину $1-v_j$ обозначим через P_j^+, а за уменьшение на величину v_j – через P_j^-.

На примере вычисления P_j^+ приведем алгоритм определения штрафных оценок при использовании для матриц B^{-1}, обратных к базисным, мультипликативного представления [3]. Для небазисных переменных зададим два списка номеров: $I_x^- = \{i: x_i = \alpha_i\}$ и $I_x^+ = \{i: x_i = \beta_i\}$. Пусть переменная x_j занимает в базисе p-ю позицию, тогда вычислим $z = e_p^T B^{-1}$, e_p – p-й орт в R^m:

$$P_j^+ = \min\left\{ \min_{q:\, q\in I_x^-,\, a_{pq}<0} \left\{ (1-v_j)\left(\frac{d_q}{-a_{pq}}\right) \right\}, \min_{q:\, q\in I_x^+,\, a_{pq}>0} \left\{ (1-v_j)\left(\frac{-d_q}{a_{pq}}\right) \right\} \right\},$$

где $a_{pq} = (z, A_q)$, A_q – q-й небазисный столбец матрицы A; d_q – его приведенная оценка $(d_q = (y, A_q) - c_q)$. Вектор y оптимальных двойственных переменных оценочной задачи ЛП и переменные d_q уже вычислены при решении этой задачи. Если q-я небазисная переменная должна быть целочисленной, то величина $(1-v_j)\left(\frac{d_q}{-a_{pq}}\right)\left(\text{или}\ (1-v_j)\left(\frac{-d_q}{a_{pq}}\right)\right)$, которая меньше $|d_q|$, заменяется на $|d_q|$. Начальные значения P_j^- и P_j^+ полагаются равными $+\infty$, и некоторые из них таковыми и останутся, если соответствующая задача ЛП не имеет допустимого решения. Однако конечное значение штрафа не означает, что отвечающая этому штрафу задача будет иметь допустимое решение.

Схемы метода ветвей и границ с односторонним ветвлением позволяют использовать компактную форму списков оценочных задач. Пусть на некотором уровне k для ветвления выбрана переменная x_j со штрафами P_j^- и P_j^+. Если $P_j^- \le P_j^+$, то в качестве очередной оценочной задачи выбирается задача, соответствующая P_j^-, а в списке оценочных задач (у нас во вспомогательном массиве h) необходимо запомнить информацию об альтернативной задаче. Для этого производятся присвоения: $h(1,k) = j$; $h(2,k) = \beta_j^i$, где β_j^i – верхняя граница переменной x_j в i-й оценочной задаче на предшествующем уровне $(k-1)$; $h(3,k) = 1$, если $f^i - P_j^+ \le r^i$ и $h(3,k) = 0$ – в противном случае. Здесь f^i – оптимальное значение f в i-й оценочной задаче, а r^i – текущее значение рекорда (если $h(3,k) = 1$, то будем говорить, что соответствующая ветвь помечена); $h(4,k) = f^i$; $h(5,k) = P_j^+$. При $P_j^+ < P_j^-$ выбирается оценочная задача, отвечающая штрафу P_j^+, а в массиве h запоминаются величины: $h(1,k) = -j$; $h(2,k) = \alpha_j^i$; $h(3,k)$ равен 0 либо 1 в зависимости от выполнения неравенства $f^i - P_j^- \le r^i$; $h(4,k) = f^i$; $h(5,k) = P_j^-$.

Алгоритм.

Шаг 0. Положить $i = 0$, $k = 0$, значение рекорда $r^0 = -\infty$, $\alpha^0 = \alpha$ и $\beta^0 = \beta$.

Шаг 1. Решить текущую задачу ЛП. Пусть x^i – ее оптимальное решение и $f^i = f(x^i)$:

а) если $f^i \le r^i$ или система ограничений несовместна, то присвоить $r^{i+1} = r^i$, $i = i+1$, и перейти на шаг 3;

б) пусть $f^i > r^i$. Если вектор x^i нецелочисленный, то перейти на шаг 2. Если решение x^i целочисленное, то присвоить $r^{i+1} = f^i$, $i = i+1$. При $f^i = f^0$ перейти на шаг 5, иначе на шаг 3.

Шаг 2. Для тех базисных переменных x_j, $j \in J$, у которых x_j^i не целое, вычислить штрафы P_j^+, P_j^-. Среди них найти минимальный штраф $P_{\min}$:

а) если $f^i - P_{\min} \le r^i$, то перейти к шагу 3;

б) при $f^i - P_{\min} > r^i$ осуществить ветвление по базисной переменной x_j, которой соответствует максимальный штраф P_j^+ или P_j^-. Из величин P_j^+, P_j^- выбрать меньшую. Пусть $P_j^- \le P_j^+$, тогда выбрать очередную оценочную задачу, соответствующую P_j^-. Увеличить $k = k+1$, записать в список задачу, отвечающую штрафу P_j^+. Изменить верхнюю границу переменной x_j, положив ее равной $[x_j^i]$. Перейти на шаг 1.

(Если $P_j^+ < P_j^-$, то в списке h запоминается задача, отвечающая P_j^-, а в формируемой задаче нижняя граница переменной x_j полагается равной $[x_j^i]+1$.)

Шаг 3. Если $k = 0$, то перейти на шаг 5.

При $k > 0$ проверить, если $h(3,k) = 1$ или $h(4,k) - h(5,k) \le r^i$, то перейти на шаг 4.

Иначе, используя массив h, сформировать задачу, альтернативную образованной в шаге 2б. Присвоить $h(3,k) = 1$ и перейти на шаг 1.

Шаг 4. Установить двусторонние ограничения на переменную x_j, используя значения $h(1,k)$, $h(2,k)$. Присвоить $k = k-1$ и перейти на шаг 3.

Шаг 5. Остановиться.

В вырожденном случае, когда максимум среди всех P_j^- и P_j^+ равен 0, для ветвления выбирается переменная x_j, которая имеет значение, наиболее уклоняющееся от целочисленного.

Выделим два случая, которые возникают при формировании оценочных задач:

1) переменной x_j предписывается фиксированное значение $x_j = \alpha_j^i = \beta_j^i$;

2) переменная должна изменяться в пределах $\alpha_j^i \le x_j \le \beta_j^i$ и $\alpha_j^i < \beta_j^i$.

В первом случае в базис предшествующей задачи с базисной матрицей B вместо переменной x_j вводится фиктивная переменная x_ν. Для этого вычисляется вектор $z^T = e_p^T B^{-1}$, где e_p – p-й орт R^m, а p – номер переменной x_j в базисе. Находится номер $k = \arg\max\{|z_i| : i = 1, \ldots, m\}$ и полагается $\nu = n + k$. Переменной x_ν ставится в соответствие k-й орт R^m. Затем обычным образом корректируется матрица B^{-1} с учетом замены в базисе переменной x_j на x_ν и вычисляются значения базисных переменных в новом базисе.

Во втором случае нет необходимости в корректировке B^{-1}. Решение очередной оценочной задачи всегда начинается с проверки выполнения для базисных переменных ограничений $\alpha_j^i \le x_j \le \beta_j^i$. Устранение недопустимости базисных переменных производится решением специальной задачи ЛП по минимизации кусочно-линейной функции

$$\phi(x) = \sum_j (\max\{0, \alpha_j^i - x_j\} + \max\{0, x_j - \beta_j^i\}),$$

где суммирование производится по базисным переменным, а для фиктивных переменных допустимыми считаются только нулевые значения.

Некоторые особенности реализации основных процедур ЛП и структуры хранения разреженных матриц и мультипликаторов рассмотрены в [4].

2. **Параллельный алгоритм.** Параллельный алгоритм устроен так, что на каждом из процессорных элементов исполняется рассмотренный ранее алгоритм с небольшими изменениями. Среди процессорных элементов выделяется элемент с нулевым номером. Данные о задаче считываются с дисковой памяти нулевым процессором и пересылаются всем остальным. Далее на нулевом процессоре решается исходная задача без учета условий целочисленности, а остальные процессоры в это время находятся в ожидании. В процессе решения на нулевом элементе накапливается справочная информация, необходимая для организации параллельных вычислений.

Для загрузки любого процессора (в дальнейшем и нулевого) в его оперативную память пересылаются массивы *ISB*, *ISN* и часть массива h. Целочисленный массив *ISB* из m компонент содержит номера базисных переменных, целочисленный массив *ISN* размера n для небазисных переменных указывает, на верхней или нижней границе находятся переменные, а для базисных переменных в *ISN* записываются номера позиций в базисе. Какая часть массива h пересылается, определяется уровнем, с которого начинается решение оценочных задач.

Пусть из нулевого процессора необходимо загрузить некоторый процессорный элемент. Предположим, что на уровне k для ветвления на нулевом процессоре выбрана переменная x_j и $P_j^- < P_j^+$. На нулевом процессоре будет решаться оценочная задача, соответствующая штрафу P_j^-. Если $f^i - P_j^+ > r^i$, то происходит передача данных на намеченный процессорный элемент. На нулевом процессорном элементе ветка, соответствующая P_j^+, помечается ($h(3, k)$ присваивается 1).

На принимающей стороне помечаются все непомеченные ветки, которые имеют уровень более высокий, нежели k. На основании списка *ISB* строится матрица B^{-1} с помощью процедуры перепостроения матриц, обратных базисным (эту процедуру называют еще повторением). Данных *ISB*, *ISN* и h достаточно для формирования первой оценочной задачи на данном процессорном элементе. Далее выполняется алгоритм ветвей и границ с односторонним ветвлением.

В параллельном алгоритме желательно на каждом процессорном элементе начинать вычислительный процесс по возможности с более высокой ветви. Для этого на каждом из процессорных элементов запоминаются массивы *ISB* и *ISN*, которые передаются некоторому другому процессору, как только поступает запрос.

Если на некотором процессоре получен новый рекорд r^i, то эта информация передается всем остальным. После получения r^i проверяется выполнение неравенства $h(4,k)-h(5,k)\le r^i$, где k соответствует уровню, с которого начиналось выполнение алгоритма на данном процессорном элементе. В тех случаях, когда это неравенство выполняется, запрашивается новое задание.

Полученное значение r^i используется для ревизии помеченных и непомеченных ветвей в массиве h. Может оказаться, что информация массивов *ISB*, *ISN* стала неактуальной (при новом значении r^i помечается ветвь, соответствующая этим массивам). В этом случае подготавливаются новые экземпляры *ISB* и *ISN* и на нулевой процессорный элемент посылается сообщение об изменении уровня, соотнесенного с этими массивами.

Задача решена, если получен рекорд $r^i = f^0$ или на всех процессорных элементах достигнут нулевой уровень.

Передача исходных данных о задаче производится с помощью блокированной функции MPI «один – всем» [5]. Дальше все процессы выполняются в асинхронном режиме и обмены осуществляются с помощью неблокированных функций.

3. **Решение задач.** В качестве тестовых взяты задачи из разных областей применения дискретной оптимизации [6]. В табл. 1 указаны входные параметры задач.

Число итераций и другие характеристики вычислительного процесса в значительной мере зависят от выбора начальных значений управляющих параметров, при этом малые изменения параметров могут привести к большим изменениям характеристик. Общее свойство неустойчивости для целочисленного линейного программирования в отдельных задачах усугубляется: неединственностью решений оценочных задач, плохой обусловленностью базисных матриц. При решении задач выбирались одни и те же начальные

Таблица 1

№ п/п	Название задачи	m	m_e	n	n_i	n_b	nz
1	air03	124	124	10757	10757	10757	91028
2	bell3a	123	–	133	71	39	347
3	bell5	91	–	104	58	30	266
4	blend2	274	89	353	264	231	1409
5	dcmulti	290	78	548	75	75	1315
6	gen	780	150	870	150	144	2592
7	gesa3	1368	48	1152	384	216	4944
8	l152lav	97	96	1989	1989	1989	9922

Примечание. № п/п – номер задачи: m – общее число ограничений, среди которых m_e ограничений являются равенствами: n – число переменных, среди которых n_i переменных являются целочисленными и n_b переменных – булевыми: nz – число ненулей в матрице A.

Таблица 2

№ п/п	Параллельный алгоритм						Один процессор		t_1/t
	$\frac{it_0}{re_0}$	$\frac{it_1}{re_1}$	$\frac{it_2}{re_2}$	$\frac{it_3}{re_3}$	$\frac{s(it)}{s(re)}$	t	$\frac{it}{re}$	t_1	
1	$\frac{1421}{28}$	$\frac{468}{10}$	$\frac{0}{0}$	$\frac{0}{0}$	$\frac{1889}{38}$	19,58	$\frac{1421}{28}$	17,04	0,87
2	$\frac{140675}{3075}$	$\frac{138439}{3122}$	$\frac{138245}{3091}$	$\frac{138624}{3074}$	$\frac{555983}{12362}$	64,06	$\frac{552937}{12154}$	251,03	3,92
3	$\frac{473266}{38581}$	$\frac{468836}{38668}$	$\frac{519309}{34402}$	$\frac{519050}{36369}$	$\frac{1980461}{148020}$	206,72	$\frac{8105266}{579981}$	3283,0	15,88
4	$\frac{41205}{902}$	$\frac{40588}{957}$	$\frac{39584}{923}$	$\frac{40781}{950}$	$\frac{162158}{3732}$	38,71	$\frac{305445}{6772}$	281,56	7,27
5	$\frac{68906}{1378}$	$\frac{53846}{1147}$	$\frac{47463}{1013}$	$\frac{48759}{1016}$	$\frac{218974}{4554}$	84,8	$\frac{390673}{7813}$	446,93	5,27
6	$\frac{42795}{858}$	$\frac{34692}{702}$	$\frac{31914}{653}$	$\frac{27926}{588}$	$\frac{137327}{2801}$	146,05	$\frac{118468}{2374}$	409,46	2,80
7	$\frac{92309}{1840}$	$\frac{49356}{1088}$	$\frac{42829}{960}$	$\frac{55231}{1185}$	$\frac{239725}{5073}$	529,28	$\frac{257365}{5132}$	1465,0	2,77
8	$\frac{463937}{9287}$	$\frac{430534}{8622}$	$\frac{456976}{9156}$	$\frac{426948}{8547}$	$\frac{1778395}{35612}$	1071,0	$\frac{2502135}{50064}$	5185,0	4,84

Примечание. № п/п – номер задачи; it_i – число итераций, выполненных на i-м процессорном элементе; re_i – число повторений на i-м элементе; $s(it)$ – суммарное число итераций на всех процессорных элементах; $s(re)$ – суммарное число повторений; t – время выполнения параллельного алгоритма (с); it – число итераций в однопроцессорном режиме; re – число повторений в последовательном алгоритме; t_1 – время решения задачи в однопроцессорном режиме (с)

значения управляющих параметров как для параллельного варианта расчетов, так и в расчетах в однопроцессорном режиме.

В табл. 2 приведены результаты расчетов. Расчеты в параллельном варианте произведены в режиме, когда каждый процесс выполнялся на отдельном процессорном элементе. Использовались четыре процессора. Как видно из данных табл. 2, задача 1, хотя и имеет высокую размерность, легко решается с помощью рассмотренного последовательного алгоритма, а в параллельном варианте не позволяет равномерно загрузить все процессоры. Задача 3 (которая имеет отношение к проектированию волоконно-оптических сетей) есть пример небольшой труднорешаемой задачи для последовательных алгоритмов.

СПИСОК ЛИТЕРАТУРЫ

1. **Муртаф Б.** Современное линейное программирование. Теория и практика. М.: Мир, 1984.
2. **Ковалев М. М.** Дискретная оптимизация (целочисленное программирование). Минск: Изд-во БГУ, 1977.
3. **Forrest J. J. H., Tomlin J. A.** Updated triangular factors of the basis to maintain sparsity in the product form simplex method // Math. Programm. 1972. **2**, N 3. P. 263.
4. **Забиняко Г. И.** Пакет программ целочисленного линейного программирования // Дискрет. анализ и исслед. операций. 1999. Сер. 2. **6**, № 2. С. 32.
5. **Корнеев В. Д.** Параллельное программирование в MPI. Новосибирск: Изд-во СО РАН, 2000.
6. **ftp**://plato.la.asu.edu

Институт вычислительной математики
и математической геофизики СО РАН,
E-mail: zabin@rav.sscc.ru

Поступила в редакцию
7 августа 2000 г.