Автоматика и телемеханика, № 5, 2007, стр. 46–58

PACS 89.20.Ff

©2007 г. Ю.Г. ЕВТУШЕНКО, академик РАН,
В.У. МАЛКОВА,
А.А. СТАНЕВИЧЮС, канд. экон. наук
(Вычислительный центр им. А.А. Дородницына РАН)

РАСПАРАЛЛЕЛИВАНИЕ ПРОЦЕССА ПОИСКА ГЛОБАЛЬНОГО ЭКСТРЕМУМА[1]

Разработан параллельный алгоритм поиска глобального экстремума функции многих переменных. Алгоритм основан на методе неравномерных покрытий, предложенном Ю.Г. Евтушенко для функций, удовлетворяющих условию Липшица. Алгоритм реализован на языке C в MPI-системе параллельного программирования с передачей сообщений. Для ускорения расчетов используются вспомогательные процедуры поиска локального экстремума. Работа алгоритма демонстрируется на примере расчета строения атомного кластера.

1. Введение

В подавляющем большинстве оптимизационных задач требуется найти глобальный оптимум. Однако из-за большой сложности поиска глобального экстремума обычно ограничиваются отысканием локальных решений. Современные многопроцессорные ЭВМ существенно расширяют возможности решения задач глобальной оптимизации, позволяя использовать параллельные алгоритмы расчетов. Перенос последовательных алгоритмов и создание новых параллельных методов глобальной оптимизации становятся весьма актуальными. Обзор многочисленных публикаций по поиску глобального экстремума приведен в [1]. Проблемы распараллеливания вычислений подробно изложены в [2].

В 1971 г. в [3] был предложен и на языке АЛГОЛ-60 реализован метод неравномерных покрытий для нахождения глобального экстремума липшицевых функций. Дальнейшее развитие этот метод получил в [4, 5]. В данной работе рассматривается параллельный вариант метода неравномерных покрытий, дается информация о программной реализации его на языке C в MPI-системе параллельного программирования с передачей сообщений (Message Passing Interface). Для ускорения расчетов используются вспомогательные процедуры поиска локального экстремума. Работа алгоритма демонстрируется на примере расчета строения атомного кластера. Вычислительные эксперименты проведены на вычислительных комплексах MVS 6000 и MVS 15000 [6].

---

[1] Работа выполнена при финансовой поддержке программы N° 14 фундаментальных исследований Президиума РАН "Раздел II: Высокопроизводительные вычисления и многопроцессорные системы" 2006 г., а также программы N° 15 Президиума РАН "Исследование и создание научных приложений в распределенной среде суперкомпьютерных вычислений на базе ВЦ РАН" 2006 г.

2. Постановка задачи, общая идея метода покрытий

Рассмотрим задачу отыскания глобального минимума функции f, определенной и непрерывной на компактном множестве $P \subset \mathbb{R}^n$

$$f_* = \operatorname{glob} \min_{x \in P} f(x). \tag{1}$$

Через X_* обозначим множество решений задачи (1), через f_* — минимальное значение целевой функции $f(x)$. Введем множество ε-оптимальных (приближенных) решений задачи (1):

$$X_*^\varepsilon = \{x \in P : f(x) \leq f_* + \varepsilon\}. \tag{2}$$

Очевидно, $X_* \subset X_*^\varepsilon \subset P$. В большинстве практических задач достаточно найти хотя бы одну точку $x_r \in X_*^\varepsilon$ и взять в качестве оптимального значения f_* число $f_r = f(x_r)$. Другими словами, надо с заданной точностью ε определить величину глобального минимума функции и найти хотя бы одну точку x_r, где это приближенное значение достигается.

Введем набор множеств $B_m = [P_1, P_2, \ldots, P_m]$ из $\mathbb{R}^n$ и набор n-мерных точек $N_m = [c_1, c_2, \ldots, c_m]$ таких, что $c_i \in P_i \subset P$ для всех $1 \leq i \leq m$. Объединение m множеств P_i обозначим через

$$U_m = \bigcup_{i=1}^{m} P_i.$$

Будем говорить, что объединение множеств P_i покрывает множество P, если

$$P = U_m. \tag{3}$$

В каждой точке c_i вычисляется значение функции f, и по формуле

$$R_m = \min_{1 \leq i \leq m} f(c_i) = f(c_r) \tag{4}$$

определяется рекордная точка c_r и рекорд R_m.

Предположим, что функция f и множество P_i таковы, что для всех $x \in P_i$ выполнено условие

$$f(x) \geq R_m - \varepsilon. \tag{5}$$

Основой метода неравномерных покрытий является следующее легко проверяемое

Утверждение 1. *Пусть набор множеств B_m и функция f таковы, что $U_m = P$ и выполнено условие* (5) *для всех $1 \leq i \leq m$. Тогда рекордная точка $c_r \in X_*^\varepsilon$.*

Действительно, если набор множеств P_i полностью покрывает P, то каждая точка x_* из X_* будет принадлежать по крайней мере одному из множеств P_i. Пусть $x_* \in P_s$, $s \leq m$; тогда, учитывая (5), получим

$$f_* = f(x_*) \geq R_m - \varepsilon = f(c_r) - \varepsilon.$$

Согласно (2), рекордная точка c_r принадлежит множеству ε-решений задачи (1).

Данное утверждение выражает основную идею метода неравномерных покрытий: вместо поиска глобального минимума на P ищутся глобальные минимумы на подмножествах, объединение которых совпадает с P. Множества P_i могут быть самыми разнообразными. Например, на каждом P_i функция f может удовлетворять условию Липшица со своей константой L_i. В некоторых областях функция может оказаться выпуклой или дважды дифференцируемой. Все эти особенности целесообразно учитывать для ускорения расчетов.

При реализации метода наборы B_m и N_m строятся последовательно, после каждого вычисления значения функции f происходит уточнение рекорда. Считаем, что последнее вычисление f было выполнено в точке c_m, рекордная точка была c_r, а рекорд был R_m. Определим следующее лебегово множество:

$$\mathcal{L}_m = \{x \in P : R_m - \varepsilon \leq f(x)\}. \tag{6}$$

Пусть глобальный минимум на $\mathcal{L}_m$ достигается в точке x_ℓ. Тогда $R_m - f(x_\ell) = f(c_r)- -f(x_\ell) \leq \varepsilon$ и, следовательно, в результате глобальной минимизации на $\mathcal{L}_m$ рекорд R_m может быть уточнен не более чем на ε. Поэтому множество $\mathcal{L}_m$ не представляет интереса и может быть исключено из области поиска P.

Если после описанных вычислений условие $P = U_m$ не выполнено, то поиск минимума продолжается на множестве $W_m = P \backslash U_m$. Пусть определены $c_{m+1} \in P_{m+1} \subseteq W_m$, вычислены $f(c_{m+1})$ и R_{m+1}. Если на P_{m+1} выполнено условие $f(x) \geq R_{m+1} - \varepsilon$, то множество P_{m+1} может быть исключено и поиск минимума продолжается на $W_m \backslash P_{m+1}$. В противном случае множество P_{m+1} разбивается на несколько подмножеств, в каждом из них вычисляются значения функции f, пересчитывается рекорд, некоторые подмножества исключаются, другие дробятся и т. д. Процесс заканчивается, когда все множество P будет полностью покрыто. Последовательность рекордов — монотонно убывающая. Последовательность лебеговых множеств $\mathcal{L}_m$, напротив, расширяющаяся, т. е. $\mathcal{L} \subseteq \mathcal{L}_{m+k}$ при $k \geq 0$. Поэтому если при некотором $s \leq m$ из P было исключено множество P_s, то P_s может быть исключено из P и при всех $s \geq m$.

Лебегово множество $\mathcal{L}$ будет наибольшим, если в (6) взять $R_m = f_*$. Однако априори величина f_* не известна. Поэтому для уменьшения рекорда целесообразно использовать методы локального поиска минимума функции f на множестве P. В тех точках $c_i \in P$, где получено, что $f(c_i) < R_m$, происходит улучшение рекорда, и есть основание полагать, что решая задачу локального поиска

$$\operatorname{loc} \min_{x \in P} f(x),$$

стартуя из точки c_i, удастся найти другую точку $\bar{c} \in P$, в которой $f(\bar{c}) < f(c_i)$. Такой прием часто существенно ускоряет расчеты, позволяя улучшить рекорд и тем самым расширить область, которую можно отбрасывать в процессе покрытия множества P.

Наиболее трудным этапом в реализации алгоритма поиска экстремума является определение хотя бы некоторой части множества $\mathcal{L}_m$. Здесь приходится налагать на функцию f дополнительные требования. Предположим, что для $f(x)$ можно построить миноранту $g(c_i, x)$ такую, что для всех $x, c_i \in P$ выполнены неравенства

$$g(c_i, x) \leq f(x), \qquad g(c_i, c_i) = f(c_i). \tag{7}$$

Определим множество

$$N_i = \{x \in P : R_m - \varepsilon \leq g(c_i, x)\}. \tag{8}$$

Из (7) следует, что $N_i \subseteq \mathcal{L}_m$, поэтому множество N_i можно исключить из области поиска P. Приведем два простейших случая, когда миноранты, удовлетворяющие (7), легко строятся. Пусть функция f удовлетворяет условию Липшица с константой L, т. е. для любых x и c_i из P выполнено неравенство

$$|f(x) - f(c_i)| \leq L \|x - c_i\|_\infty. \tag{9}$$

Здесь и ниже используется p-я гельдеровская норма

$$\|v\|_p = \left(\sum_{j=1}^{n} |v^j|^p\right)^{1/p},$$

которая при $p = \infty$ совпадает с чебышевской нормой

$$\|v\|_\infty = \max_{1 \le j \le n} |v^j|.$$

Тогда для всех $x \in P$ имеем

$$f(x) \ge f(c_i) - L\|x - c_i\|_\infty.$$

Поэтому можно взять удовлетворяющую (7) миноранту

$$g(c_i, x) = f(c_i) - L\|x - c_i\|_\infty. \tag{10}$$

Введем множество

$$K_i = \{x \in \mathbb{R}^n : \|x - c_i\|_\infty \le \rho_{im}\}, \tag{11}$$

где

$$\rho_{im} = (f(c_i) - R_m + \varepsilon)/L. \tag{12}$$

Здесь K_i — куб с центром в точке c_i и главной диагональю $2\rho_{im}$ (или если $p = \infty$, то шар с центром в точке c_i и радиусом ρ_{im}).

Если $c_i \in P$ для всех $1 \le i \le m$, то согласно (5) $f(c_i) \ge f(c_r)$, поэтому радиус ρ_{im}, задаваемый формулой (12), будет больше или равен ε/L:

$$\min_{x \in K_i} g(c_i, x) = f(c_i) - L \max_{x \in K_i} \|x - c_i\|_\infty = f(c_i) - L\rho_{im} \ge R_m - \varepsilon.$$

Следовательно, куб K_i можно исключить из области поиска P. Если объединение кубов K_i, $1 \le i \le m$, удовлетворяющих (10), покрывает множество P, то $c_r \in X_*^\varepsilon$.

Радиус шара K_i будет большим, если $f(c_i) \gg f(c_r)$, и малым в областях, где $f(c_i) \approx \approx f(c_r)$. Благодаря этому возможно неравномерное покрытие множества P шарами различного радиуса.

Если, помимо (9), выполнено условие вида $\|f'(x) - f'(z)\|_1 \le M\|x - z\|_\infty$ и σ есть скалярное произведение векторов $f'(c_i)$ и $x - c_i$, то

$$f(x) - f(c_i) \ge \sigma - M\|x - c_i\|_\infty \ge -\|x - c_i\| \cdot \|f'(c_i)\|_1 - M\|x - c_i\|_\infty^2.$$

Объединяя это с (10), получим, что минорантой будет

$$g(c_i, x) = f(c_i) - \|x - c_i\| \cdot \min\{L, \|f'(c_i)\|_1 + M\|x - c_i\|_\infty\}.$$

Возможны другие способы определения нижней оценки функции f.

3. Алгоритм последовательного покрытия

При программной реализации метода вводятся дополнительные упрощающие предположения. Считается, что допустимое множество P — это n-мерный параллелепипед с гранями, параллельными координатным плоскостям:

$$P = \{x \in \mathbb{R}^n, a \le x \le b\}.$$

Здесь и ниже неравенство $a \le x$ означает, что $a^j \le x^j$ для всех $1 \le j \le n$.

В процессе расчетов используются вспомогательные векторы $a_i, b_i \in \mathbb{R}^n$ и порождаемые ими прямоугольные параллелепипеды с гранями, параллельными координатным плоскостям: $P_i = \{x \in \mathbb{R}^n : a_i \le x \le b_i\}$.

Будем считать, что все векторы $a_i \ge a$ и $b_i \le b$. Таким образом, все параллелепипеды $P_i \subseteq P$. В качестве точек c_i берутся центры параллелепипедов P_i:

$$c_i^j = (a_i^j + b_i^j)/2, \qquad 1 \le j \le n.$$

Вектор главной диагонали d_i параллелепипеда P_i имеет компоненты $d_i^j = b_i^j - a_i^j$, $1 \le$ $\le j \le n$.

Считаем, что функция f удовлетворяет условию Липшица (9) всюду на P с одной и той же константой L. Воспользуемся минорантой (10). Обозначим

$$y_i = \min_{x \in P_i} g(c_i, x) = f(c_i) - \frac{L}{2} \max_{1 \le j \le n} \|d_i^j\| = f(c_i) - \frac{L}{2} \|d_i\|_\infty. \tag{13}$$

Если $y_i \ge R_m - \varepsilon$, то параллелепипед P_i может быть исключен, так как глобальный минимум на нем не дает улучшения текущего рекорда R_m более чем на ε, и поиск продолжается на множестве $P \backslash P_i$. В противном случае если $y_i < R_m - \varepsilon$, то параллелепипед P_i делится пополам по максимальному ребру и поиск ведется на этих двух частях. Если главные диагонали одной или обеих частей меньше, чем $2\varepsilon/L$, то эти части исключаются из области поиска. В центрах оставшихся частей вычисляются значения $f(x)$. При этом, если возможно, улучшается рекорд и проверяется условие покрытия обеих частей. Части, которые оказались покрытыми, исключаются из области поиска. Если обе части исключены, то P_i исключен, если нет, то продолжается дальнейшее деление до тех пор, пока параллелепипед P_i не будет покрыт. Возможны самые разнообразные способы покрытия параллелепипеда P. В [3], например, с помощью рекурсивных процедур на языке АЛГОЛ-60 реализовано послойное покрытие. Этот вариант использовался для решения задач глобальной оптимизации в случае, когда $n \le 15$. Ниже, следуя [4], для покрытия P применим метод ветвей и границ. Аналогичный подход для задач ранцевого типа использовался в [7].

С каждым параллелепипедом P_i свяжем набор $S_i = (c_i, d_i, y_i)$. Совокупность наборов S_i для всего набора непокрытых параллелепипедов B_m будем называть списком наборов и обозначать $S = \{S_1, S_2, \ldots, S_m\}$. Запись $S = \emptyset$ означает, что список S пуст.

В следующей формулировке алгоритма буква k будет обозначать порядковый номер итерации основного цикла.

Начальные операции. Положить $k = 1$, $m = 1$, $P_1 = P$. Задать $\varepsilon > 0$, L и некоторую точку $x_0 \in P$. Вычислить c_1, d_1, $f(c_1)$, $f(x_0)$, y_1, $z = \varepsilon/L$, $\tilde{R} = \min\{f(c_1), f(x_0)\}$. Положить $N_1^{(1)} = \{c_1\}$, $B_1^{(1)} = \{P_1\}$, $S_1 = (c_1, d_1, y_1)$, $S^{(1)} = \{S_1\}$. Если $\tilde{R} = f(c_1)$, то в качестве рекордной точки взять c_1, иначе — точку x_0. Если $y_1 \ge \tilde{R} - \varepsilon$, то закончить работу.

Основной цикл. Повторять следующие шаги 1–5, пока $S^{(k)} \ne \emptyset$, т. е. до тех пор, пока множество P не будет полностью покрыто.

1. Из текущего набора параллелепипедов $B_m^{(k)}$ выбрать тот параллелепипед P_s, для которого

$$y_s = \min_{1 \le i \le m} y_i.$$

2. В параллелепипеде P_s определить t — номер наибольшего ребра:

$$d_s^t = \max_{1 \le j \le n} d_s^j.$$

3. Разделить параллелепипед P_s пополам по t-й координате, породив тем самым два новых параллелепипеда P' и P''. Их центры и главные диагонали обозначим c', d' и c'', d'' соответственно. Исключить параллелепипед P_s из набора $B_m^{(k)}$, удалить набор S_s из списка $S^{(k)}$ и положить $m = m - 1$.

4. Вычислить

$$\tilde{R} = \min\{f(c'), f(c'')\} \tag{14}$$

и, используя (12), определить $y' = f(c') - Ld'/2$, $y'' = f(c'') - Ld''/2$.

Если $\tilde{R} \geq R^{(k)}$, то положить $R^{(k+1)} = R^{(k)}$. Если $y' < R^{(k+1)} - \varepsilon$ и $d' \geq 2z$, то включить $S' = (c', d', y')$ в список наборов и положить $m = m + 1$; если $y'' < R^{(k+1)} - \varepsilon$ и $d'' \geq z$, то включить $S'' = (c'', d'', y'')$ в список наборов и положить $m = m + 1$.

Если $\tilde{R} < R^{(k)}$, то положить $R^{(k+1)} = \tilde{R}$. В качестве рекордной точки x_r взять ту из точек c', c'', в которой достигается минимум (14). Каждый набор S_i из $\{S_i\}_{1 \leq i \leq m}$, удовлетворяющий условию $y_i \geq R^{(k+1)} - \varepsilon$, исключить из этого списка. Новый список наборов $\{S_{i1}, S_{i2}, \ldots, S_{im}\}$ перенумеровать и обозначить $S^{(k+1)} = \{S_j\}_{1 \leq j \leq m}$.

5. Положить $k = k + 1$.

Заметим, что главная диагональ параллелепипеда, полученного описанным дроблением, не меньше, чем $2\varepsilon/L$. С учетом этого факта несложно доказать, что имеет место

Утверждение 2. *Пусть в задаче* (1) *функция f удовлетворяет условию Липшица с константой L на n-мерном прямоугольном параллелепипеде P. Тогда описанный алгоритм за конечное число вычислений значений функции f определяет рекордную точку $x_r \in X_*^\varepsilon$.*

После завершения расчетов множество P будет полностью покрыто параллелепипедами P_i, а рекордная точка x_r будет ε-решением задачи (1). Можно использовать другие, отличающиеся от указанного на шаге 1, правила выбора параллелепипеда из текущего набора. Укажем следующие два правила:

- выбрать тот параллелепипед P_s, который был включен первым в текущий набор параллелепипедов;
- выбрать тот параллелепипед P_s, который был включен последним в текущий набор параллелепипедов.

Работу описанного алгоритма существенно ускоряют процедуры локального поиска экстремума. В приведенных ниже результатах расчетов для локального поиска использовался метод сопряженных градиентов с операцией проектирования на параллелепипед P. Программная реализация была осуществлена Н.И. Грачевым.

Приведенный алгоритм можно интерпретировать как метод ветвей и границ [7]:

- на шаге 1 осуществляется выбор множества для ветвления;
- на шаге 2 реализуется правило, определяющее способ ветвления;
- на шаге 3 выполняется ветвление;
- на шаге 4 по формуле (13) вычисляются оценки y', y'', отвечающие новым множествам-кандидатам для ветвления, выполняется пересчет рекорда R и производится (если рекорд улучшен) отсев множеств, которые могут быть исключены из области поиска.

На каждом шаге описанного выше метода один параллелепипед из текущего набора разбивается пополам на два новых по наибольшему ребру плоскостью, параллельной координатной. Процесс половинных делений параллелепипедов можно интерпретировать как рост двоичного дерева (рис. 1). Вершинам дерева соответствуют параллелепипеды, полученные в результате разбиения. Дуги соединяют данный параллелепипед с параллелепипедами, полученными из него в результате деления.

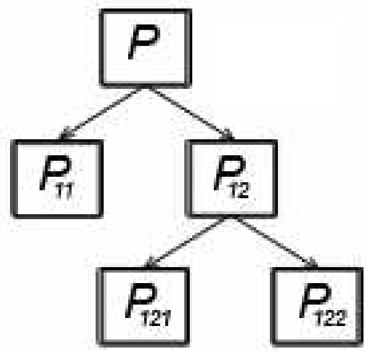

Рис. 1. Дерево ветвления для метода неравномерных покрытий.

Параллелепипеды, отвечающие концевым вершинам (P_{11}, P_{121}, P_{122}) дерева, образуют текущий набор параллелепипедов. Некоторые концевые вершины могут быть исключены из этого набора согласно правилу отсева (см. шаг 4).

Ради краткости будем далее называть текущий набор параллелепипедов пулом, а действия основного цикла, описываемые шагами 1–5, — итерацией.

4. Описание параллельного алгоритма

Главная идея параллельного алгоритма состоит в том, чтобы выполнять итерации параллельно, с периодическим обменом рекордами и перераспределением областей поиска между процессорами. Для параллельного выполнения итераций нужен способ распределения параллелепипедов из пулов между процессорами.

Сначала рассмотрим вариант алгоритма, в котором отсутствуют передачи параллелепипедов из одного пула в другой. Предположим, что для расчетов имеется p процессоров. Простейший вариант распараллеливания состоит в следующем: разобьем P на p частей, получим набор параллелепипедов P_i, удовлетворяющий условию (3). Каждый i-й параллелепипед передадим i-му процессору, $1 \leq i \leq p$. Каждому i-му процессору дадим задание: используя последовательный алгоритм, описанный в предыдущем разделе, найти на множестве P_i глобальный минимум функции f. После того как все процессоры выполнят это задание, выберем наилучший из полученных p рекордов. Этот рекорд будет ε-решением исходной задачи. Привлекательна алгоритмическая простота такого подхода. Вместе с тем он имеет ряд существенных недостатков. Укажем их.

- Время счета на различных процессорах может сильно отличаться. В результате типичной является ситуация, когда одни процессоры быстро завершают расчеты, а затем простаивают, тогда как другие, напротив, постоянно заняты расчетами. Время счета определяется тем процессором, который работает дольше всех.

- Из-за отсутствия обмена информацией полученные рекорды не передаются другим процессорам, тем самым упускается возможность ускорить расчеты.

- Отсутствует возможность подключить простаивающие процессоры и разгрузить занятые.

Рассмотрим вариант параллельного алгоритма, в котором периодически осуществляется обмен рекордами и перераспределение областей поиска между процессорами в процессе счета.

Пусть имеется некоторая начальная последовательность $B_m = \{P_1, P_2, \ldots, P_m\}$ параллелепипедов P_i, принадлежащих P, например построенная в процессе работы последовательного алгоритма. Пусть $N_m = \{c_1, c_2, \ldots, c_m\}$ — последовательность центров этих параллелепипедов. Пусть по (4) вычислены текущий рекорд R_m и текущая рекордная точка x_r.

Способ распределения работы между процессорами тесно связан с выбором единицы работы алгоритма. В качестве единицы работы определим вычислительную работу, связанную с Q-кратным $(Q \geq 1)$ выполнением итерации последовательного алгоритма.

Пусть помимо $p = m$ рабочих процессоров имеется еще один процессор, называемый диспетчером. Как и ранее, распределим параллелепипеды начального набора B_m между рабочими процессорами.

При выполнении итераций последовательного алгоритма каждый рабочий процессор поддерживает свой индивидуальный пул. Будем говорить, что наступает завершение работы процессора, если им выполнены Q циклов последовательного алгоритма (т. е. сделано Q половинных делений) или если его индивидуальный пул стал пустым. Процессор, который завершил работу, сообщает диспетчеру следующие величины: число параллелепипедов в индивидуальном пуле, минимальную оценку y для каждого из них и свой индивидуальный рекорд (т. е. лучшее значение функции f, полученное в ходе вычислений). Этот процессор ставится в поддерживаемую диспетчером очередь ожидающих процессоров. Ожидающий процессор назовем свободным, если его индивидуальный пул пуст.

Рассмотрим метод коммуникации между процессорами. Обмен информацией был бы идеальным, если каждый процессор сообщил бы другим о наилучшем найденном рекорде, о количестве имеющихся параллелепипедов и о значениях нижней оценки для параллелепипедов в текущем индивидуальном наборе. Однако слишком частые обмены между процессорами могут привести к чрезмерному увеличению коммуникационных расходов и понижению эффективности параллельного алгоритма в целом.

В предлагаемом варианте алгоритма каждый процессор посылает диспетчеру перечисленные выше данные в асинхронном режиме, т. е. независимо от других процессоров. На основе полученных данных диспетчер улучшает рекорд R и сообщает его всем ожидающим процессорам, которые будут использовать новый рекорд для отсева параллелепипедов из своих наборов. Диспетчер выявляет среди ожидающих процессоров те, которые содержат только один параллелепипед в своем индивидуальном наборе, и им приказывает выполнить основной цикл Q раз, взяв новый R в качестве рекорда. Точно такой же приказ получают другие ожидающие процессоры, если в данный момент отсутствуют свободные.

Как только диспетчер выявит свободный процессор, он среди ожидающих подберет ему партнера для передачи параллелепипеда с минимальной нижней оценкой. Подбор такой пары завершается двумя приказами: свободному процессору дается приказ принять параллелепипед от ожидающего процессора, а ожидающему — передать параллелепипед данному свободному.

Диспетчер завершает выполнение алгоритма, если все рабочие процессоры свободны.

В предлагаемом алгоритме частота обращения процессоров к диспетчеру зависит от параметра Q. Как только найдется свободный процессор, производится перераспределение вычислительной работы между процессорами путем передачи параллелепипедов. При такой схеме организации работы процессоры выполняют алгоритм асинхронно, обмениваясь информацией только при обращении к диспетчеру. В результате процесс выполнения алгоритма становится “недетерминированным”, т. е. последовательность порождаемых параллелепипедов и получаемое решение могут оказаться различными для разных запусков одной и той же задачи (даже при одном и том же наборе процессоров).

Опишем параллельный алгоритм для изложенной выше схемы взаимодействия процессоров. Для обмена данными между параллельными процессорами предполагается использовать явную передачу сообщений. Сообщение, передаваемое от рабочего процессора диспетчеру, содержит информацию о значениях следующих величин:

- число параллелепипедов в текущем пуле;
- минимальные оценки y для всех параллелепипедов текущего пула;
- индивидуальный рекорд $R = f(c)$.

Диспетчер посылает рабочим процессорам следующие сообщения-команды:

- $work(Q, R)$ — выполнить Q раз основной цикл, считая R текущим рекордом;
- $take(i)$ — принять параллелепипед от процессора i;
- $give(j)$ — передать параллелепипед процессору j;
- $finish$ — завершить работу.

Алгоритм для процессора-диспетчера:

1. Объявить свободными все рабочие процессоры, кроме одного, например p_1. Объявить процессор p_1 ожидающим и указать, что в его индивидуальном пуле единственный параллелепипед — исходный; принять рекорд $R = f(x_0)$, где $x_0 \in P$ — заданная начальная точка; положить оценку y равной оценке исходного параллелепипеда, задать параметр $Q \geq 1$.

2. Пока все процессоры не стали свободными, повторять следующие действия 3–6.

3. Если нет свободных процессоров, то каждому ожидающему дать команду $work$, а список ожидающих опустошить.

4. Каждому ожидающему процессору с единственным параллелепипедом в пуле дать команду $work(Q, R)$ и удалить из списка ожидающих.

5. (Цикл выбора пар процессоров для передачи параллелепипедов.) Пока списки ожидающих и свободных процессоров оба непусты, выбрать среди ожидающих процессор i с минимальной оценкой, а среди свободных — некоторый процессор j и удалить выбранные процессоры из соответствующих списков; отправить первому команду $give(j)$, второму — $take(i)$.

6. Принять поступающие от процессоров сообщения; занести пославший сообщение процессор в список ожидающих или свободных в зависимости от числа параллелепипедов в его пуле. Пересчитать текущий рекорд с учетом данных, полученных от процессоров.

7. Каждому процессору (они все свободны) дать команду $finish$.

Алгоритм для рабочего процессора:

1. Пока не поступила команда $finish$, выполнять действия 2–4.

2. Принять очередную команду диспетчера.

3. Если получена команда $give(j)$ или $take(i)$, то выполнить ее и отправить диспетчеру сообщение о числе своих параллелепипедов и о минимальных оценках y.

4. Если получена команда $work(Q, R)$, то выполнять итерации, пока индивидуальный пул непуст и выполнено менее Q итераций; отправить диспетчеру сообщение с результатами работы.

5. Если получена команда $finish$, то завершить работу.

5. Результаты вычислительных экспериментов

Для численных экспериментов использовалась функция, определяющая энергию атомных кластеров Морса, состоящих из n атомов:

$$f(x_1, x_2, \dots, x_n, \rho) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left[\left(e^{\rho(1-\|x_i - x_j\|)} - 1 \right)^2 - 1 \right],$$

где ρ — скалярный параметр, $x_i \in \mathbb{R}^3$ и $x_j \in \mathbb{R}^3$ — трехмерные векторы координат центров атомов i и j соответственно.

Задача состоит в определении декартовых координат n атомов таким образом, чтобы при фиксированном значении параметра ρ значение функции f достигало глобального минимума. Величина ρ принималась равной 3, 6 и 8, число атомов кластера достигало 80, при этом число искомых скалярных переменных — 240. Предполагалось, что функция f удовлетворяет условию Липшица с некоторой константой L. Так как значение L априори неизвестно, то делалось несколько расчетов: вначале с небольшим значением L, а после нахождения некоторого рекорда значение L постепенно увеличивалось таким образом, чтобы время счета было приемлемым.

Для расчетов использовался вычислительный комплекс MVS15000 Межведомственного суперкомпьютерного центра Российской академии наук [6]. Результаты вычислительных экспериментов сопоставлялись с расчетами, приведенными в Cambridge Cluster Database (базе данных по атомным кластерам Кембриджа).

Структура кластера для 25 атомов приведена на рис. 2. Значения ρ брались равными 3 и 6. Результаты расчетов с большой точностью совпали с результатами, приведенными в упомянутой базе. Максимальное значение константы Липшица было $26,3$, точность $\varepsilon = 0,1$. В определении допустимого множества P все элементы вектора a брались равными $-2,1$, а вектора b — равными $1,5$. Для локального поиска использовался метод сопряженных градиентов. Во всех случаях расчеты начинались с одной и той же начальной структуры кластера.

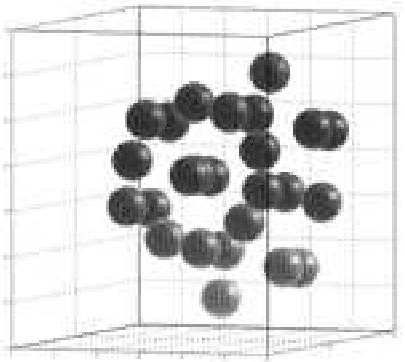

Рис. 2. Структура кластера ($n = 25, \rho = 6$).

Зависимость времени счета от параметра Q показана на рис. 3 при использовании разного числа процессоров. Расчеты показали, что наименьшее время счета достигается, если $20 \leq Q \leq 40$.

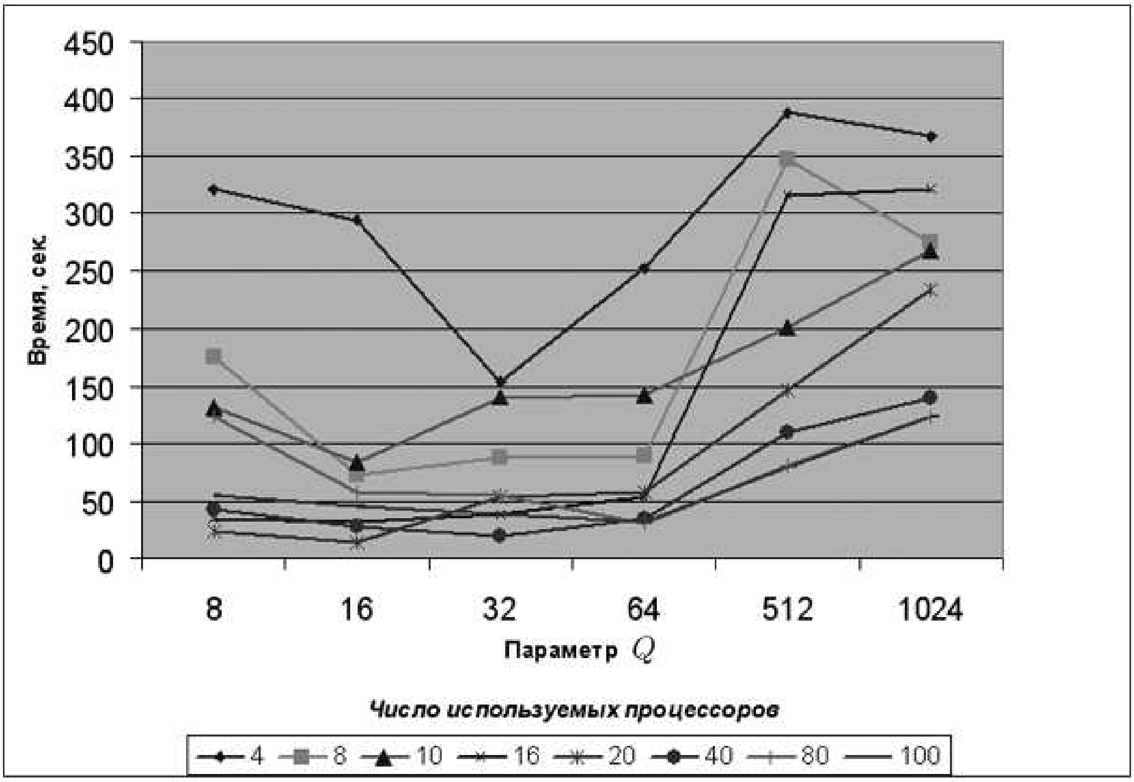

Рис. 3. Зависимость времени счета от параметра Q.

Зависимость времени счета от числа процессоров p показана на рис. 4. Для каждого числа процессоров было взято наилучшее значение параметра Q, определяемое рис. 3. Из этого рисунка следует, что время счета сокращается при увеличении числа процессоров до 20. При дальнейшем увеличении числа процессоров время не улучшается из-за увеличения простоев, неравномерности загрузки процессоров и накладных расходов, связанных с коммуникационными обменами. При $p = 4$ время счета $t = 154c$, при $p = 20$ — $t = 15c$.

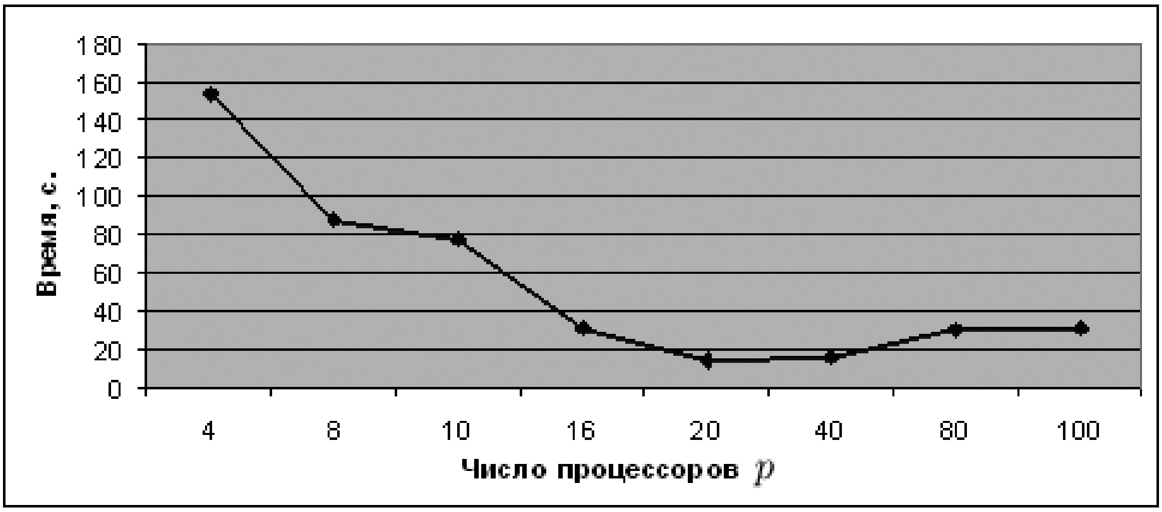

Рис. 4. Зависимость времени счета от числа процессоров.

Рисунок 5 показывает, во сколько раз по сравнению с четырьмя процессорами ускоряется счет при увеличении числа процессоров p. Так, при $p = 10$ достигается двукратное ускорение, а при $p = 20$ — десятикратное. Это ускорение получается за счет значительного сокращения числа параллелепипедов в пулах и частого обмена между процессорами информацией о текущем рекорде. Но когда число процессоров превышает 40, происходит замедление счета из-за упомянутых выше накладных расходов.

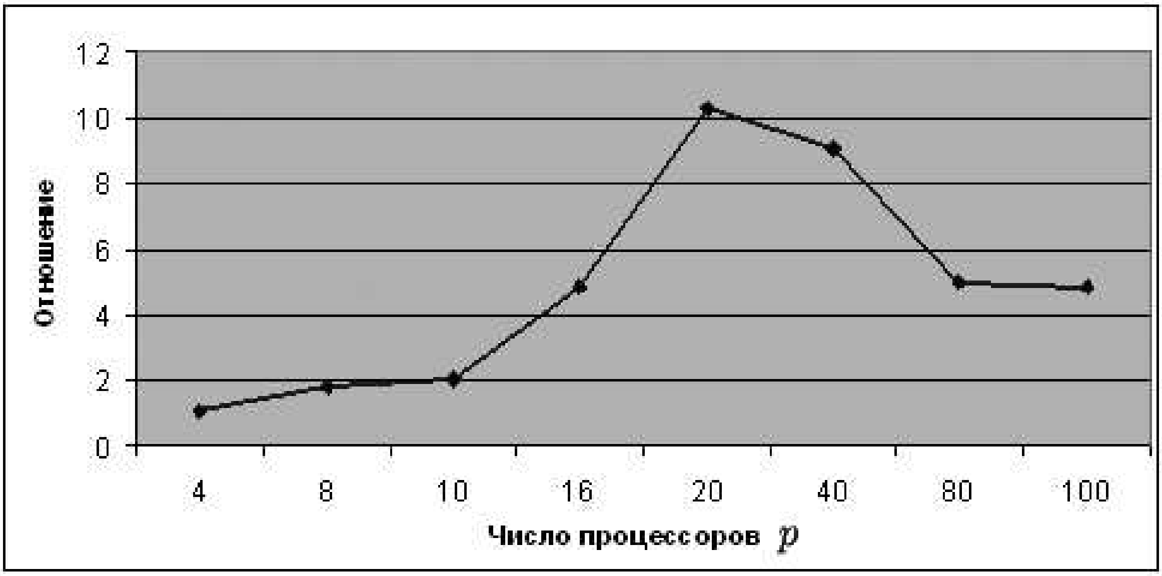

Рис. 5. Зависимость скорости счета от числа процессоров.

Рисунок 6 показывает зависимость числа всех порожденных в процессе счета параллелепипедов от числа процессоров. При $p = 100$ это число превосходило 200 000.

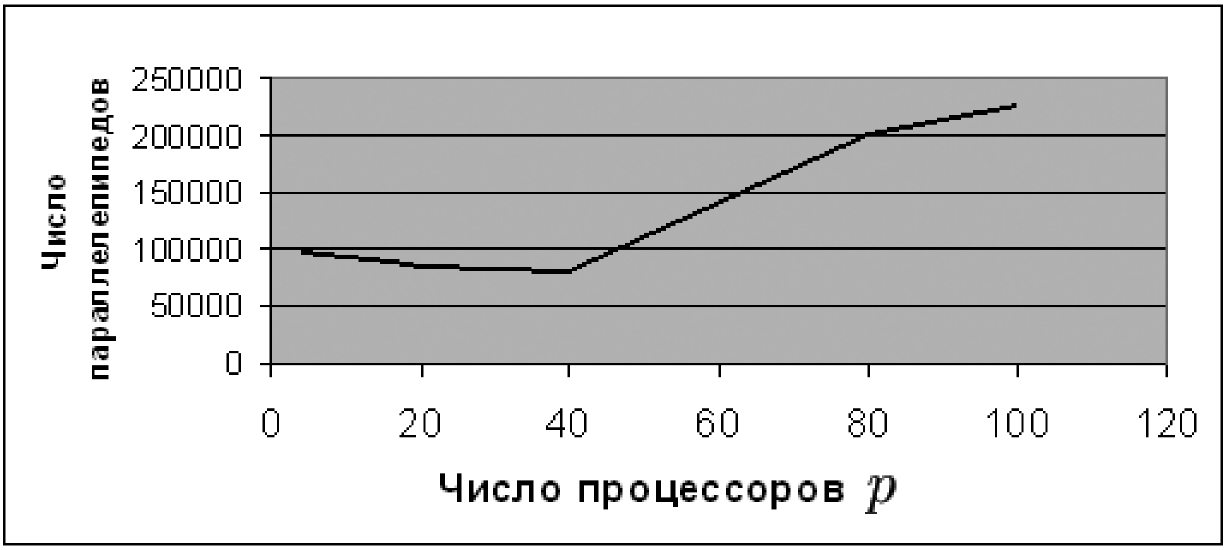

Рис. 6. Зависимость числа параллелепипедов от числа процессоров.

Для сравнения эффективности двух различных вычислительных комплексов MVS6000 и MVS15000 каждый из них использовался для расчетов оптимальной структуры кластера, содержащего 80 атомов. В обоих случаях для счета заказывалось по 50 процессоров. Максимальное время счета ограничивалось восемью часами. Был использован одинаковый набор параметров алгоритма: $\varepsilon = 0,01$, $n = 240$, $\rho = 3$, $L = 1,28$. На вычислительном комплексе MVS6000 было построено 70499 параллелепипедов и получено $f_* = -690,578$, т. е. было достигнуто значение, указанное в Кембриджской базе данных как оптимальное. Причем эта величина была получена через 5 ч 20 мин. На втором вычислительном комплексе через 5 ч 20 мин. На втором вычислительном комплексе MVS15000 за 8 ч достигнутый рекорд составил лишь $-686,306$, комплекс работал менее производительно, построив только 28653 параллелепипедов. Таким образом, для данной задачи объем вычислительной работы, проведенной первым комплексом, оказался в $2,5$ раза больше, чем вторым, вследствие чего удалось получить более точный ответ. Зависимость величины $\Delta f(x_r) = |f(x_r) - f_*|$ от времени счета на обоих комплексах приведена на рис. 7.

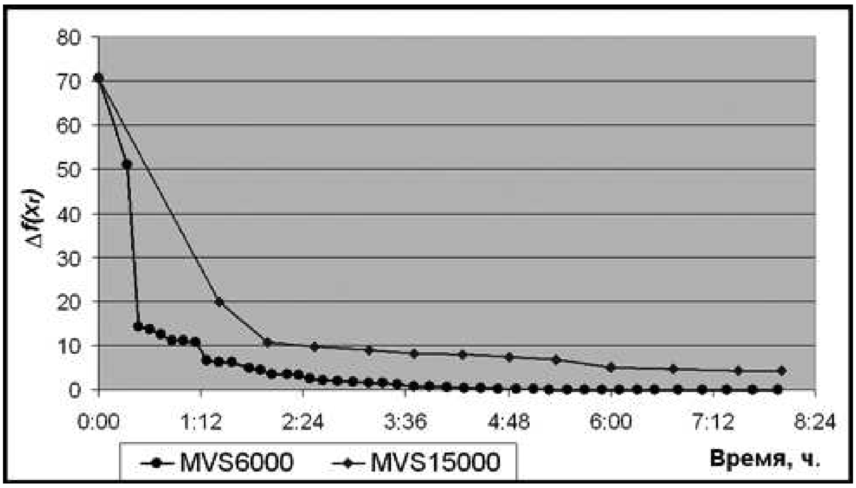

Рис. 7. Время счета на двух вычислительных комплексах.

6. Заключение

Описанный в данной статье параллельный вариант метода неравномерных покрытий позволяет проводить глобальную оптимизацию липшицевых функций в случае невысокой размерности (до 200 переменных). На эффективность распараллеливания оказывает существенное влияние не только выбор значений константы Липшица L и параметра Q (см. раздел 4), но и задание стратегии поиска глобального оптимума.

Предложенный метод параллельного глобального поиска можно применять к решению задач многокритериальной оптимизации. Наиболее простой способ состоит в использовании результатов работы [8]. Этот метод также переносится на отыскание решений задачи поиска глобального минимакса аналогично тому, как было сделано в [9]. Кроме того, его можно использовать для минимизации липшицевых функций при дополнительном требовании целочисленности всех или некоторых компонент вектора x.

Авторы выражают благодарность сотруднику ВЦ РАН А.Я. Белянкову за обсуждение статьи и сделанные полезные замечания.

Литература

1. Roman G. Strongin, Yaroslav D. Sergeyev. Global Optimization with Non-Convex Constraints. Sequential and Parallel Algorithms. Dordrecht/Boston/London: Kluwer Academic Publishers, 2000.

2. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. СПб.: BXB-Петербург, 2002.

3. Евтушенко Ю.Г. Численный метод поиска глобального экстремума функций (перебор на неравномерной сетке) // ЖВМ и МФ, 1971. Т. 11. N° 6. С. 1390–1403.

4. Евтушенко Ю.Г., Ратькин В.А. Метод половинных делений для глобальной оптимизации функции многих переменных // Техн. кибернетика. 1987. N° 1. С. 119–127.

5. Белянков А.Я. Повышение вычислительной эффективности методов неравномерных покрытий в глобальной оптимизации // Методы математического программирования и программное обеспечение. Тез. докл. Свердловск: УрО АН СССР, 1989. С. 21–22.

6. Межведомственный суперкомпьютерный центр Российской академии наук. Web-сайт http://www.jscc.ru

7. Посипкин М.А., Сигал И.Х. Исследование алгоритмов параллельных вычислений в задачах дискретной оптимизации ранцевого типа // ЖВМ и МФ, 2005. Т. 45. N° 10. С. 1801–1809.

8. Евтушенко Ю.Г., Потапов М.А. Методы решения многокритериальных задач // ДАН СССР, 1986. Т. 291. N° 1. С. 25–29.

9. Евтушенко Ю.Г. Численный метод отыскания наилучших гарантированных оценок // ЖВМ и МФ, 1972. Т. 12. N° 1. С. 89–104.

Статья представлена к публикации членом редколлегии А.И. Кибзуном.
Поступила в редакцию 18.09.2006.