МЕТОД МОНТЕ-КАРЛО В ПАРАЛЛЕЛЬНОЙ ПРОГРАММЕ МИНИМИЗАЦИИ

Иванов А.Г. [1]

Рассматривается задача минимизации скалярной функции многих переменных:

$$f(x_1, x_2, \dots x_n) \to \min .$$

При этом предполагается что значение функции $f(\cdot)$ вычисляется не по явным формулам, а является результатом счета некоторой подпрограммы. Для поиска минимума такой функции имеет смысл применять методы прямого поиска (производные функции не доступны для прямого вычисления, а при численном вычислении производных мы получаем одну из разновидностей метода прямого поиска).

В качестве основы для параллельного алгоритма был взят метод Хука-Дживса [1]. Метод состоит из последовательности шагов исследующего поиска по базовой точке и шагов поиска по образцу. Алгоритм метода является существенно последовательным: координаты каждой последующей точки становятся известными только после получения значения текущей точки. Для распараллеливания был изменен алгоритм поиска вокруг базовой точки. Разработано три варианта алгоритма различающиеся степенью перебора.

Алгоритм 1 ближе остальных к исходному алгоритму Хука-Дживса: в нем варьируются координаты базовой точки по координатным осям, число вычисляемых точек при однократном поиске вокруг базовой точки равно $2n$ (в оригинальном методе от n до $2n$), где n — размерность минимизируемой функции. В этом методе поиск по образцу совпадает с алгоритмом поиска вокруг базовой точки.

Алгоритм 2 отличается большей степенью перебора: перебираются вершины гиперкуба с центром в базовой точке, число точек при однократном поиске равно 2^n. Поиск по образцу в этом алгоритме совпадает с поиском по образцу в оригинальном методе Хука-Дживса и не распараллеливался.

[1]Работа выполнена при поддержке Российского фонда фундаментальных исследований, грант Г99–07–90441.

Алгоритм 3 реализует огромный перебор: перебираются вершины гиперкуба и середины всех его граней меньших размерностей, число точек равно $3^n - 1$. Поиск по образцу, как и в алгоритме 2, совпвдает с оригинальным.

Подробно алгоритм распараллеливания и результаты изложены в [2].

В конце поиска вокруг базовой точки существует момент, когда задача-мастер простаивает в ожидании прихода результатов от рабочих. В этот момент удобно ввести в алгоритм элементы метода Монте-Карло: вычисление случайной точки и обработка результатов счета наравне с другими точками (см. [2]).

Рассматривались различные варианты вычисления случайной точки. В результате численных экспериментов был выбран следующий вариант: случайная точка генерируется на границе гиперкуба, окружающего базовую точку, вычисление случайных точек прекращается, когда большинство рабочих уже вернули результаты.

Результаты сравнения времени счета тестовой задачи на десяти процессорах для исходного алгоритма (норма) и для алгоритма с элементами метода Монте-Карло (ММК12) приведены в таблице

параметр	N	$T_{\text{ср}}$	T_{min}	T_{max}	σ
норма	15	808.3	781.8	839.0	16.8
ММК12	21	725.9	677.0	753.2	15.3

Параметры: N - число запусков, $T_{\text{ср}}$ – среднее время счета, T_{min}, T_{max} – минимальное и максимальное времена счета, σ – среднеквадратичное отклонение.

Литература

[1]. *Банди Б.* Методы оптимизации. Вводный курс. - М.: Радио и связь, 1988.

[2]. *Иванов А. Г.* Параллельный алгоритм прямого поиска минимума функции многих переменных[2] // Алгоритмы и программные средства параллельных вычислений. - Екатеринбург: УрО РАН, Вып. 2, 1998. С. 110 – 122.

[2] Электронный вариант статьи размещен по адресу:
`http://dcs.imm.uran.ru/~u0100/IAGSoft/trasbo2_2_.html` .