

ИСПОЛЬЗОВАНИЕ ПАРАЛЛЕЛЬНЫХ ХАРАКТЕРИСТИЧЕСКИХ АЛГОРИТМОВ ДЛЯ РЕШЕНИЯ МНОГОМЕРНЫХ ЗАДАЧ ГЛОБАЛЬНОЙ ОПТИМИЗАЦИИ

К.А. Баркалов

Нижегородский государственный университет им. Н.И. Лобачевского

Введение

Рассматриваются задачи многоэкстремальной (глобальной) оптимизации вида

$$\begin{aligned}\varphi(y^*) &= \min\{\varphi(y) : y \in D\}, \\ D &= \{y \in R^N : a_i \leq y_i \leq b_i, 1 \leq i \leq N\}.\end{aligned}\tag{1}$$

Важный в прикладном отношении подкласс задач (1) характеризуется тем, что минимизируемая функция задана некоторым (программно реализуемым) алгоритмом вычисления значений $\varphi(y)$ в точках области поиска D . В этом случае об аналитические способы решения не применимы, а численное решение задачи сводится к построению оценки точного решения на основе некоторого числа k значений функций задачи, вычисленных в точках области D . Процесс вычисления значения функции в точке будем называть испытанием.

В задачах многоэкстремальной оптимизации возможность достоверной оценки глобального оптимума принципиально основана на наличии априорной информации о функции, позволяющей связать возможные значения минимизируемой функции с известными значениями в точках осуществленных поисковых итераций. Весьма часто такая априорная информация о задаче (1) представляется в виде предположения, что целевая функция φ удовлетворяет условию Липшица с константой L . Методам решения задач данного класса посвящена обширная литература (см. обзоры в [1–3]). Обзор прикладных проблем глобальной оптимизации приведен в [4].

Использование современных многопроцессорных вычислительных систем расширяет границы применения методов глобальной оптимизации и в то же время ставит задачу эффективного распараллеливания процесса поиска. Отметим, что распараллеливание процесса поиска можно проводить в нескольких направлениях. Во-первых, можно распараллеливать вычисление целевой функции, во-вторых, реализацию вычислительных правил алгоритма, обеспечивающих выбор точки проведения очередного испытания, и, в-третьих, можно изменить схему алгоритма с целью параллельного выполнения нескольких испытаний. Однако распараллеливание вычисления целевой функции, описывающей оптимизируемый объект, специфично для конкретной решаемой задачи. Распараллеливание правил алгоритма существенно зависит от конкретного класса алгоритмов и, кроме того, часто эти правила достаточно просты и распараллеливать их нецелесообразно. Поэтому перспективным является третий подход, который характеризуется эффективностью (распараллеливается именно та часть вычислительного процесса, в котором выполняется основной объем вычислений) и общностью (применим для широкого класса характеристических алгоритмов многоэкстремальной оптимизации).

Так, в работе [5] введен и исследован класс параллельных характеристических алгоритмов; установлена преемственность свойств сходимости параллельных характеристических алгоритмов по отношению к последовательным прототипам, оценена избыточность параллельных характеристических алгоритмов. Следуя этим оценкам, параллельные характеристические алгоритмы являются эффективными при небольшом числе (2–4) используемых процессоров. Использование же нескольких сотен параллельных вычислительных элементов не даст ускорения процесса поиска в несколько сотен раз. Причина состоит в том, что в этом случае параллельный алгоритм будет порождать значительное число избыточных точек испытаний (по сравнению с последовательным прототипом). В данной работе исследована новая схема использования параллельных характеристических алгоритмов, позволяющая обойти данную проблему и обеспечить значительное ускорение при решении многомерных многоэкстремальных задач.

Редукция размерности

Для снижения сложности алгоритмов глобальной оптимизации, формирующих эффективные покрытия области поиска, в многоэкстремальной оптимизации широко используются различные схемы редукции размерности, которые позволяют свести решение многомерных оптимизационных задач к семейству задач одномерной оптимизации. Редукция размерности позволяет существенно снизить сложность разрабатываемых алгоритмов глобального поиска. Кроме того, данный подход позволяет задействовать весь имеющийся аппарат одномерной многоэкстремальной оптимизации для построения эффективных многомерных методов глобального поиска.

Известно два общих метода редукции размерности. Один из них состоит в применении разверток единичного отрезка вещественной оси на гиперкуб. Роль таких разверток играют непрерывные однозначные отображения типа кривой Пеано, называемые также кривыми, заполняющими пространство. Использование развертки Пеано $y(x)$, однозначно отображающей единичный отрезок вещественной оси на единичный гиперкуб, позволяет свести многомерную задачу минимизации в области D к одномерной задаче минимизации на отрезке $[0,1]$

$$\varphi(y^*) = \varphi(y(x^*)) = \min\{\varphi(y(x)) : x \in [0,1]\}.$$

Вопросы численного построения отображений типа кривой Пеано и соответствующая теория подробно рассмотрены в [1, 3].

Второй способ редукции размерности – схема вложенной оптимизации, или же многошаговая схема – основана на соотношении

$$\min\{\varphi(y) : y \in D\} = \min_{a_1 \leq y_1 \leq b_1} \min_{a_2 \leq y_2 \leq b_2} \dots \min_{a_N \leq y_N \leq b_N} \varphi(y) \quad (2)$$

которое сводит решение исходной многомерной задачи к решению семейства рекурсивно связанных одномерных подзадач. При этом каждое вычисление целевой функции одномерной подзадачи есть решение новой одномерной подзадачи следующего уровня рекурсии, за исключением последнего уровня, где вычисляется значения исходной многомерной целевой функции. Различные модификации многошаговой схемы и соответствующая теория сходимости представлены в [3].

Для решения возникающих одномерных задач как в первом, так и во втором подходах могут быть использованы параллельные характеристические алгоритмы глобального поиска [5]. Способ распараллеливания в алгоритмах данного класса основан на следующем соображении: в характеристическом алгоритме глобального поиска каждому поисковому интервалу сопоставляется некоторое число (характеристика интервала), которое может рассматриваться как мера важности данного интервала для последующего поиска решения в нем. Следуя данному пониманию, для организации параллельных вычислений после выбора точки вычисления значения функции для первого процессора в точном соответствии с последовательным алгоритмом, для второго процессора точку очередной итерации целесообразно выбирать из следующего по важности интервала (т.е. из интервала со следующей по порядку максимальной характеристикой) и т.д.

Многоуровневая схема редукции размерности

Для изложенной выше многошаговой схемы предложено обобщение (блочная многошаговая схема), которое комбинирует использование разверток типа кривой Пеано и многошаговой схемы с целью эффективного распараллеливания вычислений.

Рассмотрим вектор y как вектор «агрегированных» макро-переменных

$$y = (y_1, y_2, \dots, y_N) = (u_1, u_2, \dots, u_M)$$

где i -я макропеременная u_i представляет собой вектор размерности N_i из последовательно взятых компонент вектора y . С использованием макро-переменных основное соотношение многошаговой схемы (2) может быть переписано в виде

$$\min_{y \in D} \varphi(y) = \min_{u_1 \in D_1} \min_{u_2 \in D_2} \dots \min_{u_M \in D_M} \varphi(y) \quad (3)$$

где подобласти D_p , $1 \leq p \leq M$, являются проекциями исходной области поиска D на подпространства, соответствующие макро-переменным u_p , $1 \leq p \leq M$.

Формулы, определяющие способ решения задачи (1) на основе соотношения (3) полностью совпадают с многошаговой схемой (2). Требуется лишь заменить исходные переменные y_p , $1 \leq p \leq N$, на макро-переменные u_p , $1 \leq p \leq M$. При этом принципиальным отличием от исходной схемы является тот факт, что в блочной многошаговой схеме вложенные подзадачи являются многомерными, и для их решения может быть применен способ редукции размерности на основе кривых Пеано. Число векторов и количество компонент в каждом векторе являются параметрами блочной многошаговой схемы и могут быть использованы для формирования подзадач с нужными свойствами, и здесь возможны два диаметрально противоположных случая.

Если $M=N$, то блочная многошаговая схема идентична исходной многошаговой схеме; каждая из вложенных подзадач является одномерной.

Если $M=1$, то решение задачи эквивалентно ее решению с использованием единственной кривой Пеано, отображающей $[0,1]$ в D ; вложенные подзадачи отсутствуют.

Нас же будет интересовать промежуточный вариант $1 < M \ll N$, при котором исходная задача большой размерности разбивается на 2-3 вложенные подзадачи меньшей размерности. Тогда, применяя в многошаговой схеме для решения вложенных подзадач параллельные характеристические методы глобальной оптимизации, мы получим параллельный вариант блочной многошаговой схемы с широкой степенью вариативности. Например, можно варьировать количество процессоров на различных уровнях оптимизации (т.е. при решении подзадач по различным макро-переменным), применять различные параллельные методы поиска на разных уровнях и т.п.

Для описания процесса параллелизма многошаговой схемы (следуя [6]) введем вектор степеней распараллеливания $\pi=(\pi_1, \pi_2, \dots, \pi_M)$, где $\pi_i, 1 \leq i \leq M$, обозначает число параллельно решаемых подзадач $(i+1)$ -го уровня вложенности, возникающих в результате выполнения параллельных итераций на i -м уровне.

В общем случае величины $\pi_i, 1 \leq i \leq M$, могут зависеть от различных параметров и меняться в процессе оптимизации, но мы ограничимся случаем, когда все компоненты вектора π постоянны. Применение параллельных характеристических алгоритмов в соответствии с блочной многошаговой схемой позволяет использовать для решения исходной задачи

$$\Pi = \prod_{i=1}^M \pi_i$$

параллельно работающих процессоров. Например, при использовании $\pi_i=8, 1 \leq i \leq M$, общее число подзадач, для которых возможно параллельное решение, будет увеличиваться на каждом уровне в 8 раз, что при использовании всего трех макропеременных составит 512. Таким образом, предложенный способ распараллеливания позволяет эффективно задействовать сотни и тысячи узлов современных вычислительных систем.

Результаты экспериментов

В данном разделе представлены результаты вычислительных экспериментов. Изложенный выше параллельный алгоритм глобальной оптимизации с использованием многоуровневой схемы редукции размерности был реализован в виде MPI-программы на языке C++. Все эксперименты были проведены на вычислительном кластере ННГУ. Формулировки и названия тестовых задач взяты из [8].

Вначале проведем исследование эффективности последовательного алгоритма на четырехмерной задаче Shekel 10. В таблице 1 приведены результаты решения данной задачи: число выполненных испытаний K , время решения T в секундах, найденное значение оптимума φ^* . Во всех экспериментах использовались параметры метода $r=2.8, \varepsilon=0.002$ и параметр построения кривой Пеано $m=10$; варьировалось лишь число M и размерность N_i макро-переменных u_i .

Таблица 1: Результаты решения задачи Shekel10

| M | N_i | K | T | φ^* |
|-----|---------|------------|-------|-------------|
| 1 | 4 | 64 604 | 0.65 | -10.53 |
| 2 | 2,2 | 140 925 | 0.89 | -10.53 |
| 2 | 3,1 | 639 780 | 3.08 | -10.52 |
| 2 | 1,3 | 399 960 | 3.44 | -10.52 |
| 3 | 1,1,2 | 1 405 598 | 8.87 | -10.52 |
| 3 | 1,2,1 | 5 863 336 | 27.7 | -10.51 |
| 3 | 2,1,1 | 3 045 636 | 14.56 | -10.52 |
| 4 | 1,1,1,1 | 18 748 757 | 88.8 | -10.52 |

Данные результаты подтверждают, что наименьшее число поисковых испытаний выполняет метод при $M=1$. Однако в данном случае метод будет плохо распараллеливаться (см. обсуждение выше), поэтому эффективность распараллеливания будем исследовать при $M=2$ и $N_i=(2,2)$. Так как значение функции вычисляется быстро, то вектор распараллеливания π будем выбирать в виде $(p-1,1)$, где p – число процессов в MPI-программе (последовательный запуск программы обозначен $(0,0)$). Ниже приведены результаты решения нескольких задач из [8]: в таблице 2 – время решения в секундах, в таблице 3 – число вычислений значений оптимизируемой функции. Также приведены используемые параметры метода r и ε и найденное значение оптимума φ^* .

Таблица 2: Результаты решения четырехмерных задач.

| Problem | r | ε | φ^* | (0,0) | (2,1) | (4,1) | (8,1) |
|------------|-----|---------------|-------------|-------|-------|-------|-------|
| Levy 1 | 2 | 0.002 | 0.0003 | 13.33 | 6.3 | 3.62 | 3.01 |
| Neumaier 2 | 2 | 0.01 | 0.005 | 31.3 | 12.1 | 9.1 | 7.38 |
| Neumaier 3 | 3.2 | 0.002 | -15.998 | 27.5 | 21.2 | 15.54 | 11.63 |
| Rastrigin | 2.5 | 0.002 | 0.003 | 2.2 | 2.96 | 1.5 | 0.7 |
| Rosenbrock | 2.1 | 0.01 | 0.1 | 14.22 | 23.31 | 9.13 | 6.7 |
| Shekel 5 | 2.5 | 0.002 | -10.152 | 0.92 | 0.35 | 0.19 | 0.18 |
| Shekel 7 | 2.5 | 0.002 | -10.402 | 0.72 | 0.44 | 0.25 | 0.16 |
| Shekel 10 | 2.5 | 0.002 | -10.534 | 0.65 | 0.54 | 0.15 | 0.23 |

Таблица 3: Результаты решения четырехмерных задач.

| Problem | r | ε | φ^* | (0,0) | (2,1) | (4,1) | (8,1) |
|---------|-----|---------------|-------------|---------|---------|---------|---------|
| Levy 1 | 2 | 0.002 | 0.0003 | 1378380 | 2145661 | 2021384 | 2349743 |

| | | | | | | | |
|------------|-----|-------|---------|---------|---------|----------|----------|
| Neumaier 2 | 2 | 0.01 | 0.005 | 3059379 | 4870091 | 6976209 | 6252963 |
| Neumaier 3 | 3.2 | 0.002 | -15.998 | 2763317 | 8636710 | 11466074 | 10176447 |
| Rastrigin | 2.5 | 0.002 | 0.003 | 235766 | 1275904 | 1286144 | 999423 |
| Rosenbrock | 2.1 | 0.01 | 0.1 | 1491435 | 9309296 | 7206413 | 9499122 |
| Shekel 5 | 2.5 | 0.002 | -10.152 | 95302 | 122831 | 105876 | 127946 |
| Shekel 7 | 2.5 | 0.002 | -10.402 | 75820 | 146677 | 135331 | 131481 |
| Shekel 10 | 2.5 | 0.002 | -10.534 | 64604 | 181132 | 71182 | 154927 |

Результаты экспериментов показывают, что последовательный алгоритм с использованием одной кривой Пеано выполняет значительно меньше поисковых испытаний, чем алгоритм с редукцией размерности по многоуровневой схеме. В силу этого в некоторых случаях получается, что параллельный алгоритм с использованием вектора распараллеливания (2,1) и, соответственно, 3 MPI-процессов работает примерно так же (или даже медленнее) чем последовательный. Однако уже использование вектора распараллеливания (4,1) дает значительное ускорение. При этом можно сказать, что по времени работы и по числу выполненных поисковых испытаний при решении данных задач предложенный алгоритм не уступает другим известным методам [7].

Следующую серию экспериментов проведем на задаче Rastrigin из [8] при $N=10$, $D=\{-2.2 \leq y_i \leq 1.8, 1 \leq i \leq N\}$. Глобально-оптимальное решение – $y^*=(0, \dots, 0)$, $\varphi(y^*)=0$. В таблице 4 приведены результаты решения данной задачи: время решения T в секундах и найденное значение оптимума φ^* . Во всех экспериментах использовались параметры метода $r=2.3$, $\varepsilon=0.01$, параметр построения кривой Пеано $m=10$. Число макропеременных и их размерность были выбраны $M=2$, и $N_i=(6,4)$, т.е. решение исходной 10-мерной задачи сводится к решению 6-мерной, причем вычисление одного значения в ней подразумевает решение вложенной 4-мерной подзадачи. Варьировался вектор распараллеливания π , в соответствии с которым изменялось число процессов в MPI-программе от 9 до 101 соответственно.

Таблица 4: Результаты решения десятимерной задачи.

| π | T | φ^* |
|---------|--------|-------------|
| (8,1) | 15 938 | 0.26 |
| (20,1) | 5 448 | 0.18 |
| (40,1) | 2 911 | 0.19 |
| (70,1) | 1 453 | 0.27 |
| (100,1) | 2 697 | 0.18 |

Во всех экспериментах число поисковых испытаний изменялось незначительно и составило порядка 10^9 . Однако при этом произошло значительное сокращение времени решения задачи за счет увеличения числа используемых процессоров.

Заключение

В данной работе исследовалась многоуровневая схема редукции размерности в задачах глобальной оптимизации, комбинирующая использование кривых Пеано и многошаговую схему. Для решения редуцированных подзадач используется алгоритм глобального поиска, принадлежащий классу характеристических алгоритмов. Для предложенной многоуровневой схемы рассмотрены вопросы ее эффективного использования для задач с разным временем вычисления оптимизируемой функции.

Проведены вычислительные эксперименты на серии тестовых задач разной размерности с целью сравнения ускорения. Результаты экспериментов показывают, что предложенная схема позволяет эффективно распараллелить процесс поиска и добиться значительного ускорения. При этом разработанный алгоритм как минимум не уступает, а в некоторых случаях и превосходит известные методы аналогичного назначения.

ЛИТЕРАТУРА:

1. R.G. Strongin, Ya.D. Sergeyev. Global optimization with non-convex constraints. Sequential and parallel algorithms. – Kluwer Academic Publishers, Dordrecht. – 2000.
2. C.A. Floudas, C.E. Gounaris. A review of recent advances in global optimization // *Journal of Global Optimization*, vol. 45, pp. 3–38, 2009.
3. Стронгин Р.Г., Гергель В.П., Гришагин В.А., Баркалов К.А. Параллельные вычисления в задачах глобальной оптимизации – М.: Издательство Московского университета. – 2013.
4. J.D. Pinter. Global Optimization: Scientific and Engineering Case Studies. – Springer Verlag. – 2006.
5. V.A. Grishagin, Ya.D. Sergeyev, R.G. Strongin, Parallel Characteristical Algorithms for Solving Problems of Global Optimization // *Journal of Global Optimization*, vol. 10, pp. 185–206, 1997.
6. Ya.D. Sergeyev, V.A. Grishagin. Parallel Asynchronous Global Search ant the Nested Optimization Scheme// *Journal of Computational Analysis and Applications*, vol. 3(2), pp. 123–245, 2001.
7. M. Ali, Ch. Khompatraporn and Z.B. Zabinsky. A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems // *Journal of Global Optimization*, vol. 31, pp. 635–672, 2005.

8. R. Paulavicius, J. Zilinskas and A. Grothey. Parallel branch and bound for global optimization with combination of Lipschitz bounds // Optimization Methods & Software, vol. 26(3), pp. 487–498, 2011.