АДАПТИВНЫЕ ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ ДЛЯ МНОГОМЕРНОЙ МНОГОЭКСТРЕМАЛЬНОЙ ОПТИМИЗАЦИИ

В.П. Гергель, В.А. Гришагин, А.В. Гергель

Решение задач многомерной многоэкстремальной оптимизации является сложной проблемой. Сложность задач подобного вида определяется, прежде всего, экспоненциальным ростом объема вычислений при увеличении размерности (числа варьируемых параметров). Другой существенный фактор состоит в том, что во многих случаях расчет значений функционалов оптимизационной задачи (целевой функции и ограничений) требует существенного объема вычислений, поскольку зачастую связан с проведением вычислительного эксперимента с той или иной математической моделью исследуемого объекта, системы или явления. Именно на такие вычислительно-трудоемкие задачи ориентирована разработка параллельных методов многомерной многоэкстремальной оптимизации в данной работе.

В качестве основного принципа построения параллельных методов многоэкстремальной оптимизации в данной работе будет применяться схема параллельных вычислений, в которой обеспечивается возможность одновременного (параллельного) расчета значений функционалов оптимизационной задачи в нескольких разных точках области поиска (см., например, [1-2]). Такой подход характеризуется *эффективностью* (распараллеливается именно та часть вычислительного процесса, в котором выполняется основной объем вычислений) и *общностью*, поскольку применим для всех вычислительно-трудоемких задач многоэкстремальной оптимизации.

Предлагаемый в работе подход будет рассмотрен применительно к широкому классу характеристически-представимых алгоритмов глобального поиска [3].

1. Задачи многомерной многоэкстремальной оптимизации и многошаговая схема редукции размерности

Задача многомерной многоэкстремальной оптимизации может быть определена как проблема поиска наименьшего значения действительной функции $\varphi(y)$

$$\varphi(y^*) = \min\{\varphi(y): y \in D\}, \tag{1}$$

где D есть область поиска, представляющая собой некоторый гиперпараллелепипед N-мерного евклидова пространства.

Многие регулярные поисковые методы решения многомерных оптимизационных задач сводят многомерную задачу (явно или неявно) к системе одномерных подзадач. Один из наиболее общих методов редукции размерности состоит в применении *многошаговой схемой редукции размерности*, согласно которой решение многомерной задачи оптимизации может быть получено посредством решения последовательности «вложенных» одномерных задач (см., например, [1-3]):

$$\min_{y\in D} \varphi(y) = \min_{y_1\in[a_1,b_1]} \cdots \min_{y_N\in[a_N,b_N]} \varphi(y_1,\ldots y_N) \tag{2}$$

Согласно (2) решение многомерной задачи (1) сводится к решению одномерной задачи:

$$\varphi^* = \min_{y\in D} \varphi(y) = \min_{y_1\in[a_1,b_1]} f_1(y_1), \tag{3}$$

где

$$f_i(y_i) = \varphi_i(y_1,\ldots,y_i) = \min_{y_{i+1}\in[a_{i+1},b_{i+1}]} \varphi_{i+1}(y_1,\ldots,y_i,y_{i+1}) \quad 1\le i<N, \tag{4}$$

$$\varphi_N(y_1,\ldots,y_N) = \varphi(y_1,\ldots,y_N). \tag{5}$$

Правила (3) – (5) определяют множество задач:

$$F_l = \{f_i(y_i),\ 1\le i\le l\}. \tag{6}$$

порождаемых в соответствии с многошаговой схемой редукции. Количество задач в множестве F_l в процессе поиска может изменяться: увеличиваться при переходе к следующей переменной и уменьшаться при завершении решения какой-либо из задач (можно отметить, что при этом количество задач не превышает размерность решаемой задачи N). При этом активной – решаемой – в множестве F_l является только одна задача – это задача с максимальным номером варьируемой переменной.

Многошаговая схема редукции размерности позволяет получить общий способ для применения одномерных алгоритмов оптимизации к решению многомерных оптимизационных задач. Вместе с этим, данной схеме присущи и определенные недостатки – так, например, часть вычислений является избыточным, поскольку решение исходной задачи оптимизации сводится к минимизации одномерных функций в раздельных подобластях области поиска.

В работе предлагается обобщенная (адаптивная) многошаговой схемы редукции размерности, в которой предлагается осуществлять одновременное решение всех задач множества F_l. Такое решение может выполняться последовательно (при наличии единственного процессора), тогда для выполнения очередной итерации глобального поиска необходимо выбирать для решения одну из задач множества F_l в соответствии с тем или иным правилом выбора задач. Но решение задач может выполняться и параллельно, если используемый вычислитель является многопроцессорным или многоядерным. Именно параллельный – более общий вариант – будет рассматриваться далее в работе.

Важно отметить, что выполнение итерации глобального поиска для любой задачи множества F_l с номером переменной меньшим, чем N (N есть размерность решаемой задачи), будет приводить к порождению новой одномерной задачи вида (4), и, тем самым, количество задач в семействе F_l может оказаться значительным (десятки и сотни тысяч для сложных задач оптимизации).

Выделим в семействе решаемых задач оптимизации F_l из (6) задачи для последней варьируемой переменной (задачи уровня N) – будем называть задачи подобного вида *терминальными* (нетерминальные задачи семейства будут именоваться далее *структурными*). В соответствии с выдвинутыми в начале работы предположениями именно при решении таких задач выполняется наибольший объем вычислений, т.к. только в этих задачах осуществляется вычисление значений функционалов исходной решаемой задачи оптимизации.

2. Централизованная схема параллельного глобального поиска

Последовательные характеристически-представимые алгоритмы обеспечивают выбор только единственного точки в интервале с максимальной характеристикой для проведения очередного испытания. Используемые в алгоритмах характеристики интервалов могут рассматриваться как некоторые меры важности интервалов на предмет содержания в интервалах искомого решения оптимизационной задачи [1-2]. Следуя данному пониманию, в параллельных алгоритмах после выбора точки проведения испытания для первого процессора в точном соответствии с последовательным алгоритмом, для второго процессора точку испытания целесообразно выбирать из следующего по важности интервала (т.е. из интервала со следующей по порядку максимальной характеристикой) и т.д.

Рассмотренная выше идея послужила основой для целого направления работ по разработке параллельных информационно-статистических алгоритмов глобального поиска – см., например, [2, 4]. Один из важных результатов в данном направлении состоит в том, что для параллельных алгоритмов оптимизации оказалось возможным формулирование общей схемы характеристической представимости [3-4]. Данная схема может быть обобщена для предлагаемого в работе подхода.

Схема характеристической представимости параллельных алгоритмов многоэкстремальной оптимизации на основе адаптивной многошаговой редукции размерности. Пусть $p>1$ есть число процессоров, используемых для решения задачи оптимизации. Каждый из процессоров, получив точку $y \in D$, осуществляет независимо и параллельно с другими процессорами вычисление значения функции $\varphi(y)$ из (1). В процессе вычисления значения функции испытание считается незавершенным – тем самым, при выполнении алгоритма будут существовать точки завершенных и незавершенных испытаний.

Далее, пусть проводятся первые $p>1$ испытаний в некоторых точках области поиска D и количество завершенных испытаний $k=0$. Тогда выбор точки очередной итерации (*решающее правило алгоритма*) включает выполнение следующего набора действий:

1. **Начало итерации**. Если завершено $k>0$ испытаний в точках $y^1, y^2, \ldots, y^k$, и реализуются испытания в точках $y^{k+1}, y^{k+2}, \ldots, y^{k+p}$, вычислительная схема алгоритма находится в состоянии ожидания. Если же одно из параллельных испытаний завершается, номер k увеличивается на единицу и алгоритм переходит к формированию точки x^{k+p+1} нового $k+p+1$ испытания.
2. **Вычисление характеристик**. Для всех задач семейства F_l необходимо выполнить правила 2.1 – 2.4.

2.1. **Формирование системы интервалов. Координаты $x^1, x^2, \ldots, x^{s(j)}$, испытаний (завершенных и незавершенных) вместе с граничными точками a_j и b_j области одномерной задачи оптимизации $f_j(x) \in F_l$ перенумеровываются нижним индексом в порядке возрастания:**

$$a_j = x_0 \le x_1 \le x_2 \le \ldots \le x_{s-1} \le x_s = b_j \ .$$

2.2. **Список допустимых интервалов**. Сформировать список номеров I_s допустимых интервалов, в качестве которых рассматриваются такие интервалы, граничные точки которых являются координатами завершенных интервалов, либо граничными очками области $[a_j, b_j]$.

2.3. **Вычисление характеристик допустимых интервалов**. Каждому допустимому интервалу (x_{i-1}, x_i), $i \in I_s$, ставится в соответствие величина $R(i)$, называемая далее *характеристикой интервала*.

2.4). **Выбор интервала с максимальной характеристикой**. Среди допустимых интервалов определить интервал (x_{t-1}, x_t), $t \in I_s$, которому соответствует максимальная характеристика $R(t)$, т.е.

$$R(t) = \max\{R(i): i \in I_k\} \tag{7}$$

(данная характеристика $R(f_j) = R(t)$ будет называться также *характеристикой задачи* $f_j(x) \in F_l$).

3. **Вычисление точки нового испытания**. Точка x^{k+p+1} очередной итерации глобального поиска выбирается внутри интервала $(x_{t-1}, x_t)_k$ для задачи $f_\tau(x) \in F_l$ с максимальной характеристикой $R(f_\tau)$ в соответствии с правилом

$$x^{k+p+1} = s(t) \in (x_{t-1}, x_t), \; R(f_\tau) = \max\{R(f_j): f_j \in F_l\}. \tag{8}$$

Если выбранная задача является структурной (номер переменной не соответствует размерности задачи), то порождается новая задача семейства в соответствии с правилами (3)-(5) и выполнение правил начинается с шага 2; для терминальной задачи в соответствии с выбранной точкой x^{k+p+1} формируется вектор варьируемых переменных y^{k+p+1} для очередной испытания (вычисления значения оптимизируемой функции).

4. **Проверка условия остановки**. Завершение глобального поиска происходит, если в правиле 2.3 выбирается задача $f_\tau(x) \in F_l$, соответствующая первой варьируемой переменной, а длина интервала (x_{t-1}, x_t) с максимальной характеристикой $R(t)$. Оказывается меньшей заданной точности поиска $\varepsilon > 0$, т.е. выполняется условие

$$x_t - x_{t-1} < \varepsilon$$

Характеристическая схема представимости параллельных алгоритмов глобального поиска может быть детализирована для получения того или иного метода многоэкстремальной оптимизации указанием правила вычисления характеристики интервалов (7) и правила определения точки очередного испытания (8). Принципиально важный момент состоит в том, что в качестве этих правил могут быть использованы правила практически любого существующего последовательного алгоритма поиска глобального поиска и, как результат, на его основе может быть сформирован параллельный метод многоэкстремальной оптимизации, соответствующий исходному последовательному алгоритму. Так, в частности, могут быть задействованы правила метода последовательного сканирования, метода ломаных, а также правила всех информационно-статистических алгоритмов [3].

3. Распределенная схема параллельного глобального поиска для адаптивной многошаговой схемы редукции размерности

Централизованная схема организации параллельных вычислений является достаточно эффективной при сравнительно небольшом количестве используемых процессоров. В случае же многопроцессорных систем с существенным числом (сотни и тысячи) процессоров управляющий узел системы, обеспечивающий выполнение решающих правил алгоритмов, обработку поисковой информации и определение точек очередных испытаний может оказаться «узким» местом. Управляющий процессор может не успевать обеспечивать нагрузку для всех процессоров-исполнителей даже если этот управляющий процессор будет обладать повышенной производительностью. Все дальнейшее рассмотрение будет выполнено на примере для информационно-статистических многомерных методов многоэкстремальной оптимизации с адаптивной многошаговой схемы редукции размерности.

3.1. Общая схема распределенных вычислений

Введем множество номеров задач семейства F_l из (6):

$$L = \{1, 2, \ldots, l\}, \tag{9}$$

и пусть для проведения вычислений имеется $p > 1$ процессоров. Распределим имеющиеся задачи между процессорами – данное распределение можно зафиксировать при помощи соответствующего разделения множества L на подмножества:

$$\Pi = \{\pi_1, \pi_2, \ldots, \pi_p\}, \tag{10}$$
$$\pi_i = \{j_s : j_s \in L, 1 \le s \le l_i\}, 1 \le i \le p,$$
$$\forall i \in L \; \exists j : i \in \pi_j, \forall i,j \Rightarrow \pi_i \cap \pi_j = \{\varnothing\}$$

где π_i, $1 \le i \le p$, есть множество задач, распределенных для решения на процессоре i, $1 \le i \le p$.

Построенная схема является децентрализованной – все процессоры работают параллельно и самостоятельно генерируют точки проведения испытаний. С другой стороны, между процессорами остается информационное взаимодействие – при получении в какой-то задаче новой улучшенной оценки минимального значений оптимизируемой функции эта оценка должна передаваться задаче-родителю. Поскольку решаемые задачи теперь распределены между процессорами, то передача получаемых оценок между задачами может

приводить к сложным информационным зависимостям между процессорами. Понятно, что для минимизации таких зависимостей распределение решаемых оптимизационных задач между процессорами должно быть согласовано со структурой информационной зависимости задач.

3.2. Структурная схема распределенных вычислений

Для решения проблемы распределения задач между процессорами может быть предложена простая и эффективная децентрализованная схема распределенных вычислений, в которой в достаточной степени учитывается структура иерархической зависимости решаемых задач оптимизации.

Пусть номера терминальных задач образуют множество

$$L^N = \{ i_j : i_j \in L, 1 \le j \le l^N \}. \quad (11)$$

Распределим терминальные задачи между процессорами и зафиксируем применяемое распределение по аналогии с (10) при помощи множества:

$$\Pi^N = \{ \pi_0, \pi_1, \pi_2, \dots, \pi_p \}, \quad (12)$$
$$\pi_0 = L \setminus L^N, \pi_i = \{ j_s : j_s \in L^N, 1 \le s \le l_i \}, 1 \le i \le p,$$
$$\forall i \in L \; \exists j : i \in \pi_j, \forall i,j \Rightarrow \pi_i \cap \pi_j = \{\varnothing\}.$$

В (12) каждое подмножество π_i, $1 \le i \le p$, определяет список терминальных задач (задач уровня N), распределенных для решения на процессоре i, $1 \le i \le p$. Подмножество π_0 содержит номера всех структурных задач (задач с уровнем меньшим, чем N). Вопрос распределения задач подмножества π_0 по процессорам необходимо разрешить дополнительно; в наиболее простом варианте для задач данного подмножества можно выделить дополнительный процессор. Получаемая в результате такого решения топология многопроцессорной системы становится близкой к варианту централизованной схемы распределенных вычислений. Тем не менее, новый разработанный вариант организации вычислений по прежнему остается децентрализованным, т.к. процессоры с терминальными задачами самостоятельно, не зависимо от управляющего процессора, определяют точки проведения своих испытаний. Новая схема отличается также и систематическим характером информационных взаимодействий между процессорами. Процессоры с терминальными задачи пересылают получаемые оценки минимальных значений исходной оптимизируемой задачи только управляющему процессору. Управляющий процессор занимается обработкой только структурных задач без выполнения трудоемких вычислений значений функционалов исходной решаемой задачи оптимизации. При этом управляющий процессор может порождать терминальные задачи и, в этом случае, эти задачи должны переправляться для выполнения на те или иные процессоры с терминальными задачами.

3.3. Балансировка вычислительной нагрузки в структурной схеме распределенных вычислений

Процесс глобального поиска на основе адаптивной многошаговой схеме редукции размерности отличается высокой динамикой: постоянно осуществляется порождение новых одномерных задач оптимизации, в результате чего количество решаемых на процессорах задач может оказаться существенно различным. Кроме того, характеристики задач $R(f_j)$, $j \in L$, L из (9) могут меняться достаточно быстро, что приводит к частой смене «важных» задач (задач с максимальными характеристиками), подлежащих решению в первую очередь. Отменные моменты означают, что в процессе глобального поиска обязательным моментом является балансировка вычислительной нагрузки для процессоров. Следует отметить, что данный аспект – балансировка вычислений – является одни из наиболее важных и сложных проблем организации распределенных вычислений (см., например, [5]).

Далее последовательно будут рассмотрены два способа динамической балансировки.

12. **Глобальная балансировка**. Преобразуем множество терминальных задач L^N из (11) в список задач, упорядоченный по убыванию характеристик задач, т.е.

$$L^N = \{i_1, i_2, \dots, i_{l^N} : i_j \in L^N, 1 \le j \le l^N\} \quad (13)$$

где i_1 есть терминальная задача с максимальной характеристикой

$$i_1 = arg \max R(f_j), j \in L^N,$$

i_2 есть терминальная задача со следующей по порядку максимальной характеристикой и т.д. Тогда для максимального соответствия схеме характеристической представимости параллельных алгоритмов многоэкстремальной оптимизации (см. раздел 2) терминальные задачи из списка $\vec{L}^N$ из (12) должны распределяться между процессорами строго последовательно, т.е. задача i_1 должна быть распределена на 1 процессор, задача i_2 - на 2 процессор и т.д. При исчерпании процессоров распределение задач далее можно продолжить циклически, т.е. задача i_1+p должна быть распределена на 1 процессор, задача i_2+p - на 2 процессор и т.д. В общем виде, сформированное правило можно зафиксировать следующим образом:

Процессор j, $0 \le j \le p$, используется для решения терминальных задач
с номерами i_j, i_j+p, i_j+2p, и т.д.

Предложенное правило балансировки является практически оптимальным, поскольку в максимальной степени соответствует схеме характеристической представимости параллельных алгоритмов многоэкстремальной оптимизации, однако реальное применение данного правила затруднительно в силу существенных накладных расходов, необходимых для его реализации (высокая интенсивность передачи данных между процессорами). Предлагаемый далее способ локальной попарной балансировки требует для своего выполнения меньших вычислительных затрат.

13. **Локальная попарная балансировка**. Для проведения балансировки данного вида представим все имеющиеся процессоры (кроме управляющегося процессора) в виде набора непересекающихся пар процессоров

$$\Sigma = \{ \sigma_s=(i_s, j_s) : 1 \le s \le p/2 \}. \quad (14)$$

В предлагаемом способе локальной попарной балансировки перераспределение терминальных задач осуществляется только между процессорами образованных пар. Балансировка состоит в следующем - для каждой пары процессоров (i,j) разбиения Σ из (14) выполняются действия:

- Формируется объединенный список терминальнах задач процессорной пары:

$$\pi = \pi_i \cap \pi_j = \{ j_1, j_2, \dots, j_\tau : \tau = l_i + l_j \},$$

- Список задач упорядочивается по убыванию характеристик задач:

$$\vec{\pi} = \{ i_1, i_2, \dots, i_\tau : i_j \in \pi, 1 \le j \le s \},$$

где i_1 есть задача с максимальной характеристикой среди задач набора π, i_2 есть задача со следующей по порядку максимальной характеристикой и т.д.;

- Набор задач $\vec{\pi}$ разделяется между процессорами пары, для чего на первом процессоре пары оставляются задачи, находящиеся в списке $\vec{\pi}$ на позициях с нечетными номерами (т.е. это задачи i_1, i_3, … и т.д.), а на втором процессоре пары - задачи, находящиеся в списке $\vec{\pi}$ на позициях с четными номерами.

Для проведения локальной попарной балансировки в работе предлагаются способ чет-нечетного группирования и метод группирования процессоров с использованием топологии гиперкуба.

Работа выполнена при поддержке РФФИ (грант № 07-01-00467-а) и Совета по грантам Президента Российской Федерации по государственной поддержке ведущих научных школ Российской Федерации (грант № НШ-4694.2008.9).

Список литературы:

1. Р.Г. Стронгин . Численные методы в многоэкстремальных задачах. М.: Наука, 1978.
2. R.G. Strongin , Y.D. Sergeyev Global Optimization with non-convex constraints: Sequential and parallel algorithms. Kluwer Academic Publishers, Dordrecht, 2000.
3. С.Ю. Городецкий , В.А. Гришагин Нелинейное программирование и многоэкстремальная оптимизация. – Н.Новгород: Изд-во ННГУ, 2007.
4. Y.D. Sergeyev , V.A. Grishagin Parallel asynchronous global search and the nested optimization scheme. // J. Comput. Anal. Appl. – 2001. – Vol. 3, № 2. – P. 123-145.
5. В.П. Гергель Теория и практика параллельных вычислений. – М.: Интернет-Университет Информационных технологий; БИНОМ. Лаборатория знаний, 2007.