Параллельные алгоритмы интервальной глобальной оптимизации

М. Е. Лозбень
Новосибирский государственный университет

Н. В. Панов
Конструкторско-технологический институт вычислительной техники СО РАН

Для задачи доказательной (гарантированной) глобальной оптимизации, когда требуется не только найти оптимум функции, но и дать гарантию того, что найденное решение является действительно глобальным экстремумом, интервальный анализ предлагает мощный инструмент – интервальное расширение функции, которое позволяет находить гарантированные внешние оценки области значений функции на интервале. Это эксплуатируют интервальные методы поиска глобального оптимума, основанные на оценивании целевой функции и адаптивном дроблении области определения. Традиционно подобные методы признавались вычислительно менее эффективными, чем основанные, например, на техники распространения ограничений. Тем не менее, методы адаптивного дробления позволяют добиться эффективного распараллеливания вычислений. Это, а также применение стохастических техник, таких как интервальные генетические алгоритмы, позволяет создавать высокоэффективные параллельные интервальные алгоритмы глобальной оптимизации.

Постановка задачи и ее актуальность

Задачей оптимизации в математике называется задача о нахождении экстремума (минимума или максимума) вещественной функции в некоторой области. Как правило, рассматриваются области, принадлежащие R^n и заданные набором равенств и неравенств.

Методы оптимизации присутствуют практически на всех этапах системного анализа, а так же в системах поддержки принятия решений [1]. Оптимизация находит широкое применение в науке, технике и во многих других практических областях деятельности человека. Целевая функция может быть многоэкстремальной, разрывной, недифференцируемой, а также может быть искажена помехой.

В связи с бурным развитием вычислительной техники становятся все

более актуальными различные численные методы оптимизации, так как происходит значительное удешевление вычислительных мощностей, что позволяет использовать современные алгоритмы оптимизации для более точного решения ряда проблем, для которых ранее было возможно получение лишь грубого решения. В настоящее время параллельные и векторные суперкомпьютеры рассматриваются как один из основных инструментов для проведения исследований в различных научных и прикладных дисциплинах. Современные многопроцессорные компьютеры получили широкое распростронение, и последние исследования в оптимизации принимают во внимание архитектурные особенности современных компьютеров, на которых эти алгоритмы предполагается реализовать.

Обзор алгоритмов

Задача глобальной оптимизации зачастую сводится к задаче поиска локальных экстремумов и нахождению среди них глобального оптимума. Алгоритмы поиска локального экстремума предназначены для определения одного из локальных экстремумов на множестве допустимых решений, в котором целевая функция принимает максимальное или минимальное значение [2]. При их построении могут использоваться три типа стратегий: детерминированный спуск в область экстремума, случайный поиск, комбинированные методы, содержащие элементы детерминированного и случайного поиска.

Алгоритм поиска локального экстремума обычно содержит процедуру "условия прекращения поиска экстремума", которая обеспечивает нахождение экстремума с определенной точностью. В целом алгоритмы поиска реализуют методы спуска к экстремуму, при которых значение целевой функции последовательно улучшается вплоть до достижения экстремума.

Методы спуска разделяются на 3 группы [1, 2]:

1) методы, использующие значения 1-й и 2-й производных целевой функции В этом случае выполняются как необходимые, так и достаточные условия экстремума. Ярким представителем этих методов является метод Ньютона и его модификации;

2) методы, использующие только значения 1-й производной целевой функции, по которой можно оценить необходимое условие экстремума, а значит, выбрать верное направление спуска. К этой группе принадлежат градиентные методы, метод наискорейшего спуска, примыкают методы спуска с переменной метрикой, в которых вычисляются только 1-е производные, однако информация используется для последовательного построения матрицы 2-х производных (методы Флетчера-Ривса, Дэвидсона - Флетчера - Пауэлла и т.д.);

3) при оптимизации параметров технических систем наиболее широко применяются методы прямого поиска, не использующие вычисления производных, например, метод Гаусса-Зейделя и его модификации, метод Розенброка, метод Хука-Дживса, причем последние два гораздо эффективнее первого. Для задач со сравнительно небольшим (до 10) кол-вом переменных наиболее эффективен алгоритм поиска по деформируемому многограннику (метод Нелде-ра-Мита). В большинстве применяемых в настоящее время методов поиска экстремума при определении направления и размера последующего шага используется информация, полученная только на одном-двух предыдущих шагах. Это позволяет сократить время на использование информации, однако увеличивает общее количество шагов.

Интервальный анализ

Наш алгоритм поиска глобального оптимума основывается на интервальной арифметике и методе ветвей и границ.

Интервальная арифметика — математическая структура, которая для вещественных интервалов определяет операции, аналогичные обычным арифметическим. Развитие «интервальной идеи» состоялось лишь в XX веке, причём оно оказалось тесно связанным с развитием и распространением практических вычислений. А оформление интервального анализа в самостоятельную научную дисциплину вообще стало возможным лишь с появлением ЭВМ [3].

В 1931 году англичанка Розалинда Янг разработала арифметику для вычислений с множествами чисел. В 1951 году П. Двайер в США рассматривал специальный случай замкнутых интервалов (числовые диапазоны) в связи с необходимостью учёта погрешностей в численном анализе. В 1956-58-м годах появились работы Мечислава Вармуса в Польше и Теруо Сунаги в Японии, предлагавшие классическую интервальную арифметику и намечавшие её приложения. При этом в впервые были использованы и современные термины «интервал», «интервальный». Кроме того, Т. Сунага заложил основы интервального алгебраического формализма и дал весьма нетривиальные примеры применений новой техники, к примеру, в численном решении алгебраических уравнений и задачи Коши для обыкновенных дифференциальных уравнений.

В России и Советском Союзе «интервальную» историю можно отсчитывать с 20-х годов прошлого века, и связана она с именем замечательного русского советского математика и педагога В.М.Брадиса. С середины 20-х годов прошлого века он проповедовал так называемый метод границ – способ организации вычислений, приводящий к достоверным двусторонним границам точного значения вычисляемого результата, фактически аналогичный интервальной арифметике.

Предлагаемый нами метод глобальной оптимизации

Для решения задачи оптимизации в последние десятилетия было предложено большое количество подходов, каждый из которых имеет свои преимущества и недостатки[4]. Тем не менее, общими чертами большинства из них являются:

— локальный характер, и, как следствие, неспособность находить гарантированно глобальный оптимум целевой функции,

— гарантированные оценки точности полученных решений либо находятся подобными методами с большим трудом, либо не находятся вообще.

Наш метод глобальной оптимизации, основанный на применении интервального анализа, свободен от этих недостатков, так как способен исследовать целые куски области определения целевой функции, имеющие ненулевую меру. Кроме того, он не теряет решений-оптимумов, гарантируя нахождение оптимума в указанной области.

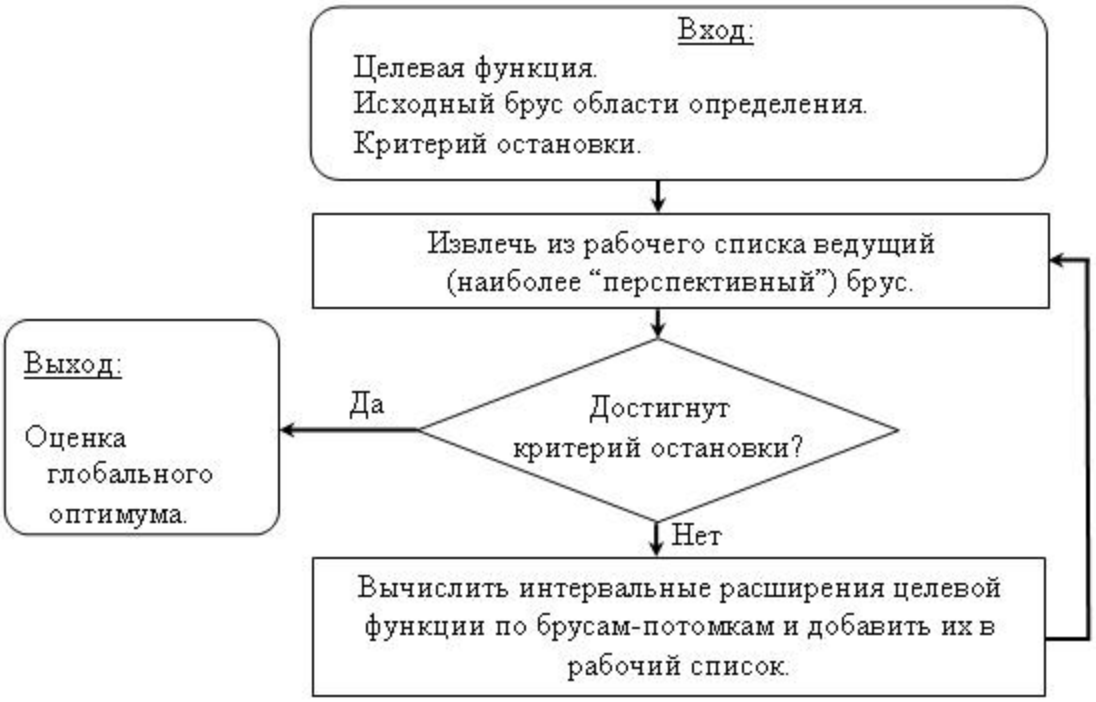
Интервальный тип данных и интервальная арифметика реализуются на современных ЭВМ, например, представлением интервала как пары чисел – одного для левого конца интервала, а другого для правого. При этом существующее аппаратное обеспечение, в частности, арифметика чисел с плавающей точкой, используются без каких-либо изменений, так как корректность получающейся интервальной арифметики может быть обеспечена так называемыми направленными округлениями. Например, там, где в задачах внешнего интервального оценивания в процессе вычислений требуется округление результата, нижняя граница интервала должна округляться вниз, а верхняя граница интервала – вверх. Таким образом даже неизбежные ошибки округления при вычислениях с плавающей точкой будут строго и систематически учитываются в процессе выполенния интервальной программы [3].

Таким образом, интервальный подход к данному классу задач представляется нам наиболее оптимальным.

Идея метода

Метод последовательно уточняет интервальную оценку целевой функции на подозрительных интервалах, сновываясь на метое "Адаптивного интервального дробления". Основная функиональнасть — планомерное дробление наиболее подозрительного на данном шаге интервала, уточняя таким образом положение и значение глобального оптимума [4].

Так выглядит блоксхема алгоритма:



Под брусом понимается многомерный интервал, исходный брус — область определения целевой функции, критерии остановки опциональны и зависят от конкретной задачи — это может быть, например, время работы, узость границ оценки оптимума, время работы программы.

Можно выделить следующие основные этапы интервального алгоритма уточнения глобального оптимума:

1. Выбрать перспективную область.
2. Раздробить выбранную область на подобласти.
3. Вычислить интервальное расширение функции на полученных подобластях.

Разница между алгоритмами в деталях. По каким критериям выбирать брус для дробления, как дробить, как вычислять интервальные расширения, какой выбрать критерий остановки [5].

Работа программы

Рассмотрим подробно работу программы. На вход подается исходный брус, целевая функция и в текущей реализации необходимая точность (в качестве критерия остановки). Далее начинается дробление области на части.

На каждой итерации алгоритма делению подвергаются наиболее «подозрительный» брус, то есть тот, на котором потенциально может достигаться экстремум. Один из способов его выбора — определение бруса, на котором целевая функция достигает наименьшего значения (имеется ввиду нижняя оценка интервального значения функции на брусе).

После выбора бруса он проверяется на соответствие критерию остановки.

В случае удовлетворения требованиям, он добавляется к остальным, а затем из всех рабочих брусов собирается область, в которой и лежиит наш глобальный оптимум. Если же нет, то выбранный брус подвергается дроблению, на полученных подбрусах считаются интервальные значения целевой функции и пони добавляются к остальным рабочим.

Есть возможность дробления на равные половины, на n частей (по одной из осей координат) равномерно или в заданной пропорции, также разрабатывается адаптивное дробление (Дробление с памятью) — дробление, на каждом шаге определяющее в каких пропорциях, на сколько частей и главное — по какой координате дробить рабочие области в зависимости от результатов предведущих итераций. В числе прочего, для этого предполагается использовать генетические алгоритмы программирования.

Неравномерное дробление имеет большой потенциал, так как исходя из специфики целевых функций позволяет подобрать наиболее эффективный метод работы для каждого их вида. Например, большинство функций монотонны на весьма обширной области определения по сравнению с размерами интервала, да которых мы дробим (ведь интересующая нас точность $1e^{-2}$ — $1e^{-6}$). Соответственно, на таком отрезке локальный минимум (максимум) будет находится на одном из концов интервала. В таком случае неравномерное дробление позволит получить более узкие границы оптимума за меньшее количество дроблений. Метод сложен в реализации: переключение с одного варианта дробления на другой, необходимость ведения статистики переходов и его анализа на каждой итерации алгоритма, поэтому пока ещё тестируется и дорабатывается.

В процессе работы программы периодически запускается процедура отбраковки брусов, на которых экстремум заведомо не достигается, отделяя лишние брусы и формируя множество «рабочих брусов». Отбраковываемые брусы определяются наличием или отсутствием брусов, на которых верхние границы интервальных значений целевой функции ниже. Действительно, если есть брус, на котором значения целевой функции достигают гарантированно меньшие значения, чем на текущем брусе (верхняя оценка функции на нем ниже нижней оценки функции на текущем), то на нем достижение минимума невозможно. Всвязи с своеобразной работой с оперативной памятью в Java, процедура запускается в тот момент, когда встроенный сборщик мусора сигнализирует о нехватке памяти, что обеспечивает ее своевременый запуск именно в нужный момент. Эта необходимость продиктована громоздкостью работы процедуры.

На данном этапе есть две параллельные реализации алгоритма. Одна из них делит исходный брус на n равномерных областей и запускает n потоков, каждый из которых обрабатывает алгоритмом соответствующую область. Затем результаты собираются вместе и, как и в линейной реализации, выделяется искомая область, в которой гарантированно находится оптимум.

Вторая более интересна. В ней реализована адаптивная параллельная

работа потоков. Есть основной процесс, контроллер, который запускает остальные и осуществляет функции координации и управления ими. На начальном этапе он формирует для каждого потока исходный брус. Также реализована возможность каждому потоку задать свой тип алгоритма, включая условия остановки, алгоритмы дробления и выбора брусов. Это позволит в дальнейшем расширить адаптивность программы с упоминавшегося выше неравномерного дробления до полной функциональной гибкости, возможности оценив статистику и проанализировав результаты предведущих итераций на отдельных областях выбрать для конкретного потока наиболее оптимальный подход, адаптируясь к специфике целевой функции.

Потоки на каждой итерации алгоритма передают контроллеру данные о состоянии работы на текущий момент: текущий минимум целевой функции, количество рабочих брусов и состояние (достигнут или нет критерий остановки для данной конкретной области). Контроллер на основании этих данных отслеживает, какова текущая граница отбраковки и нужно ли перераспределять рабочие брусы с загруженных потоков на отработавшие. Тут возникает проблема реализации эффективного межпотокового взаимодействия.

Во-первых, возникла проблема — каким именно образом хранить границы отбраковки, в какой момент вычислять глобальную для всех потоков границу, когда ее передавать и как синхронизировать отбраковку брусов на разных потоках? Выяснилось, что стандартное решение — запись в одну глобальную переменную первым потоком своей границы и последующее сравненивание с ней значений границ остальных потоков приводит к ошибкам, так как пока один поток читает ее для сравнения, другой сразу после этого ее перезаписывает. Возникает так называемая ошибка параллельной модификации данных (date race). Если, для решения этой проблемы, ввести блокировку потоков при обращении к этой переменной на чтение или запись (синхронизованный доступ), то это приведёт к тому, что всем n потокам придётся ждать своей очереди для сравнения (так как переменная при работе с одним из них для остальных становится недоступна). Что на корню убьёт идею параллельной работы нескольких алгоритмов. Обозначенную проблему можно решить реализовав неблакирующую конкурентную запись максимального значения в общую глобальную переменну, однако на вычислительных машинах с большим количеством сокетов это привело бы к дополнительным синхронизациям кэша каждого из ядер и основной памяти. Была предложена альтернативная реализация — выделить каждому потоку свою переменную для записи в ней текущего состояния и предоставить контроллеру находить из них наименьшую и передавать её всем в качестве эталона. При таком подходе тоже есть свои проблемы — каждому потоку приходится ждать, когда все остальные запишут свои значения и контроллер выполнит свою работу, передав эталонное значение ему и всем остальным. Кроме того, контроллеру приходится отслеживать момент, когда все потоки записали свои границы. А так как это происходит на каждой итерации, то серьёзно снижает скорость работы в целом. В ближайшее время планируется подробные исследование связанных вопросов

на многопроцессорных системах для выбора оптимальной реализации.

Ссылки и литература

1. Кузнецов Алексей Владимирович. *Алгоритмы глобальной оптимизации функций в пространстве непрерывных переменных при наличии ограничений-неравенств* – диссертация на соискания степени к.ф.-м.н., Красноярск, 2006.
2. Жиглявский А.А., Жилинскас А.Г. *Методы поиска глобального экстремума.* – М.: Наука, 1991.
3. Шарый С.П. *Конечномерный интервальный анализ*. – Электронная книга, доступная на http://www.nsc.ru/interval/index.php?j=Library/InteBooks/index
4. Панов Н.В. Объединение стохастических и интервальных подходов для решения задач глобальной оптимизации функций. // *Вычислительные технологии.* – 2009. – Т. 14, №5. – С. 49 – 65.
5. Панов Н.В. Адаптивный мета-алгоритм глобальной оптимизации // *Естественные и технические науки*. – 2009, № 1 (39). – М.: Спутник+, С. 315 – 318.