Обзор методов глобальной оптимизации

Постановка общей задачи глобальной оптимизации

Без ограничения общности, рассмотрим задачу глобальной минимизации:

где многоэкстремальная функция цели является невыпуклой и для различных алгоритмов имеет различные свойства гладкости, - допустимое множество задачи, - **k**-мерное Евклидово пространство.

Множества

,

множество оптимальных решений и множество локально-оптимальных решений соответственно и . Таким образом, задача ГО состоит в отыскании точки , причем в случае, когда .

В силу широты класса многоэкстремальных функций задача ГО в общем случае является неразрешимой, т.е. нельзя гарантировать, что решение задачи будет получено за конечное число шагов. В работе [49] была получена нижняя оценка стохастической сложности для многоэкстремальной задачи оптимизации *k* раз непрерывно дифференцируемой функции *n* переменных

, (1)

где *ε* – заданная погрешность решения оптимизационной задачи по координате, *c(n,k)* – некоторый коэффициент, зависящий от свойств функции.

В работе [50] для задачи безусловной глобальной оптимизации гладкой липшицевой функции была получена более оптимистическая оценка сложности:

. (2)

Тем не менее, для решения задачи глобальной оптимизации функции 100 переменных для гладких липшицевых функций потребуется приблизительно обращений к целевой функции. Из выражения (1) и (2) ясно, что при заданной точности увеличение размерности задачи оптимизации приводит к катастрофическому росту ее сложности. Даже некоторые из разрешимых задач могут попасть в класс неразрешимых, если число шагов, необходимых для получения решения, окажется чрезмерно большим. Для того чтобы решить задачу, помимо непрерывности функции цели, необходимы некоторые дополнительные условия ее гладкости. Таким образом, специфика задачи ГО заключается в многоэкстремальности функции цели и неразрешимости в общем случае.

В настоящей работе основное внимание уделяется безусловным задачам глобальной оптимизации, т.е. задачам, в которых глобальное решение достигается во внутренней точке допустимого множества [1], [2], [3]. В связи с этим предполагается простая структура допустимого множества, обычно это многомерный гиперкуб, без ограничения общности, единичный .

Для получения точного решения безусловных задач ГО было предложено множество методов. Точные методы либо полагаются на априорную информацию о том, насколько быстро изменяется функция (например информация о константе Липшица функции), либо требуют аналитически сформулировать функцию цели (например, в методе интервалов). Статистические методы, как правило, используют технику разбиения, чтобы разделить область поиска, но такие методы также пользуются априорной информацией или некоторыми предположениями о том, как функция цели может быть промоделирована. Основное предположение заключается в том, что каждая конкретная целевая функция принадлежит классу функций, которые промоделированы конкретной стохастической функцией. Информация из предыдущей выборки целевой функции может быть использована для оценки параметров стохастической функции, и эта усовершенствованная модель впоследствии может быть использована для смещения при отборе точек в исследуемой области.

Одновременно с возникающими задачами и потребностями в их решении ведутся работы по разработке новых и усовершенствованию существующих специальных инструментов, позволяющих решать эти проблемы, используя весь накопленный опыт и современные вычислительные средства. Проблемам ГО, разработке новых методов и другим важным исследованиям в ГО посвящено немало работ. Интернет-сайт по проблемам ГО - <http://www.mat.univie.ac.at/~neum/glopt.html>

Свойства методов глобальной оптимизации

Процедуру (или алгоритм) решения задачи оптимизации можно представить в виде итеративного процесса, который порождает последовательность точек в соответствии с предписанным набором правил, включающим критерий окончания счета. Обычно, глобальное решение задачи оптимизации предлагается найти, перебрав все ее локальные решения. Такая задача, как правило, оказывается трудоемкой. Другой подход - перебрать часть локальных решений и показать, что оставшиеся локальные минимумы не влияют на точность решения. Таким образом, идея всех методов ГО - оценить значения целевой функции на некотором множестве точек *x1,..xN* из допустимого множества *X*, и различие методов заключается в способах выбора этих точек.

Поскольку мы не знаем, где именно в *X* можно найти глобальные минимумы, то необходимо применить некоторую стратегию, чтобы «разбросать» эти точки по множеству *X*. Любую такую стратегию называют глобальной техникой. Далее, вполне возможно, что в окрестности выбранной точки *xn*существует лучшее значение функции цели, чем . Для получения лучшего значения функции к полученной точке *xn* применяют алгоритм локального спуска. Метод, осуществляющий такой локальный спуск, называют *локальной техникой*. Почти все методы ГО используют методы локальной оптимизации, по крайней мере, для того чтобы улучшить уже найденную оценку глобального решения, поэтому независимо от применяемой глобальной техники, использование локальной техники является важной частью любого метода ГО.

Другим важным свойством алгоритма оптимизации является сходимость генерируемой им последовательности точек к глобальному оптимальному решению. Однако в большинстве случаев получаются менее благоприятные результаты: невыпуклость функций, большая размерность задачи или другие трудности вынуждают останавливать алгоритм, если получена точка, принадлежащая некоторому множеству приближенных решений. Другими словами, для любого численного алгоритма необходимы условия остановки. Это ключевой момент, т.к. без какой-либо дополнительной информации или предположений о задаче невозможно сделать выводы о точности решения, полученного за некоторое фиксированное число шагов алгоритма. Таким образом, для четкого определения условий остановки алгоритма необходима дополнительная информация или предположения, или же условия остановки должны быть эвристическими. Это подтверждает высказанный ранее факт, что задача ГО в общем случае неразрешима, и мы должны быть готовы принять полученное с помощью численного метода приближение за решение задачи. Таким образом, получить численное решение задачи ГО означает получить некую точку

, (3)

где

, (4)

а величина *ε* неизвестна.

Любой численный метод имеет свои преимущества и недостатки, в связи, с чем возникает вопрос о сравнении различных методов. Существует ряд факторов, которые следует учитывать при оценке эффективности алгоритмов и их сравнении. Так, универсальность алгоритма определяется тем классом задач, для решения которых он предназначен, а также рамками требований, предъявляемых алгоритмом к задачам данного класса. Другими важными характеристиками алгоритма является его надежность (или устойчивость), точность, чувствительность к параметрам и исходным данным, затраты на предварительную обработку и вычисления.

Кроме того, с появлением параллельных вычислительных систем возникает другой интересный вопрос: как сравнивать параллельный и последовательный алгоритмы. Не вызывает сомнения факт, что сравнивать параллельные алгоритмы необходимо по-своему. Для оценки эффективности параллельных алгоритмов используют различные подходы, наиболее распространенным из которых является показатель ускорения. Ускорение, получаемое при работе алгоритма на *p* процессорах - это отношение времени работы алгоритма на одном процессоре ко времени работы того же алгоритма на *p* процессорах. Закон Амдала [5] позволяет вычислить верхнюю границу ускорения, которое можно ожидать от параллельной реализации алгоритма.

Некоторые методы глобальной оптимизации

К сожалению, для решения задачи ГО не существует универсального по эффективности алгоритма. Поэтому при разработке специфических методов ГО в первую очередь учитывают свойства целевой функции и допустимого множества *X* рассматриваемого класса задач, для которых разрабатывается метод.

Все известные методы ГО можно разделить на две категории: детерминированные [6] и стохастические [7]. Детерминированные методы получают глобальное решение посредством исчерпывающего поиска на всем допустимом множестве. Поэтому большинство детерминированных методов теряют эффективность и надежность с возрастанием размерности задачи. Более того, чтобы гарантировать успех, такие методы требуют выполнения дополнительных предположений, наложенных на функцию цели. Детерминированные методы не используют стохастики.

Стохастические алгоритмы позволяют уйти от проблем детерминированных алгоритмов. Здесь стохастический подход присутствует не только в разработке и анализе алгоритма, но и используется в решении базовых проблем, например, при определении условия остановки. Большинство стохастических методов оценивают значение функции цели в случайных точках допустимого множества с последующей обработкой выборки. Как следствие, стохастические методы не гарантируют успех.

Ниже приведены некоторые примеры современных методов ГО.

Очень широкий обзор современных достижений в области ГО приведен в статье [2]. Описание методов оптимизации также можно найти в [3], [6], [10], [12], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23].

Исторически первым методом ГО является метод Монте-Карло, на базе которого был создан метод мультистарта [20]. В методе мультистарта из множества *X* случайно или детерминировано выбирается некоторое подмножество из *N* точек. Последовательно из каждой точки запускается алгоритм локального спуска, и из полученного множества локальных решений выбирается наилучшее. В чистом виде метод мультистарта не является эффективным, т.к. одно и то же локальное решение может быть найдено не один раз. Мультистарт - это обобщенный подход: большинство эффективных методов ГО основано на идее метода мультистарта - запуска стандартных локальных алгоритмов из множества точек, равномерно распределенных на множестве *X*. Таким образом, метод мультистарта можно назвать прототипом таких методов.

Методы группировок [21] являются одной из модификаций метода мультистарта. Здесь предпринята попытка устранения главного недостатка мультистарта путем тщательного отбора точек, из которых запускается локальный поиск. Рассматривается некоторая выборка точек, например, равномерно распределенных на *X*. Затем, пока не выполнится некоторое условие остановки, последовательно выполняются следующие три шага:

1. из каждой точки запускается алгоритм локального спуска, в результате будет получен набор локальных решений;
2. используя специальную технику группировки, определяются группы точек;
3. в качестве новой выборки точек рассматривается каждая *m*-тая точка из группы, и осуществляется переход к первому шагу.

Таким образом, решения находятся посредством локальных алгоритмов спуска из наилучших точек каждой группы.

Методы деления пополам и методы интервалов [1], [10], [22], [23] гарантируют, что решение будет получено с заданной точностью. Эти методы в ГО также носят название методов покрытий или методов ветвей и границ. Цена предлагаемой методами гарантии - некоторая a priori известная информация о функции цели. Так, для метода деления пополам необходимо знание константы Липшица функции цели, а для метода интервалов функция цели обязана быть дважды непрерывно дифференцируемой, и первая и вторая производные должны иметь конечное число нулей. Большинство методов интервалов используют стратегию ветвей и границ [23]. Такие алгоритмы разделяют область поиска на набор многомерных кубиков, на которых нижняя граница функции цели вычисляется с помощью интервальной техники. Используя интервальную арифметику на каждом шаге, получаем набор последовательно уменьшающихся интервалов, который содержит глобальное решение исходной задачи. Алгоритм останавливается, когда размер интервалов достигает заранее заданного значения. Методы интервалов требуют, чтобы функция цели была задана явно, т. к. это выражение используется интервальной техникой для вычисления границ. Методы, использующие технику ветвей и границ, в силу древовидной структуры являются объектом исследований в контексте параллельных вычислений. Некоторые разработки параллельных алгоритмов ветвей и границ приводятся в [23].

Эволюционные алгоритмы [25] являются поисковыми методами, основная идея которых заимствована из биологического процесса естественного отбора и процесса выживания. Такие алгоритмы отличаются от традиционных методов оптимизации тем, что поиск производится из «популяции» решений, а не из одной точки. Каждая итерация метода производит «естественный отбор», который отсеивает неподходящие решения. Решения с высокой пригодностью («биологической реакцией на естественный отбор») «скрещиваются» с другими решениями путем обмена частей одних решений на другие. Решения могут «мутировать» из-за небольших замен одного элемента решения. Скрещивания и мутации генерируют новые решения, которые «генетически» настроены на области допустимого множества, для которых уже было обнаружено хорошее решение. Существует несколько различных типов эволюционных поисковых алгоритмов: алгоритмы генетического программирования; алгоритмы эволюционного программирования; алгоритмы эволюционных стратегий; генетические алгоритмы. Эволюционные алгоритмы обладают слабой сходимостью к глобальному решению, но вместе с тем, хорошо обрабатывают сильно зашумленные функции с большим числом незначительных локальных решений, не «прилипают» к локальным экстремумам и способны получить глобальное решение.

Методы редукции размерности позволяют свести решение многомерных оптимизационных задач к семейству задач одномерной оптимизации. Один из наиболее общих методов редукции размерности состоит в применении многошаговой схемы редукции размерности [26]*,* согласно ко­торой решение многомерной задачи оптимизации может быть получено посредством решения последовательности «вложенных» одномерных задач. Другим общим способом редукции размерности основывается на из­вестном фундаментальном свойстве, согласно которому *N*-мерный гиперпараллелепипед и отрезок вещественной оси [0,1] являются равномощными множествами и отрезок мо­жет быть однозначно и непрерывно отображен на гиперпараллелепипед. Отображения такого рода обычно называют развертками или кривы­ми Пеано. В большинстве случаев в методах редукции ставка делается на эффективность одномерных алгоритмов глобальной оптимизации, например, на информационно-статистический подход Р.Г. Стронгина [43].

Оценка сложности методов глобальной оптимизации

Согласно работе [50] в таблицу 1 сведены оценки сложности задачи глобальной оптимизации функции многих переменных для разных методов ее решения и свойств самой функции.

Таблица 1 Оценка вычислительной сложности задачи глобальной оптимизации

| *n* | 2 | 3 | 4 | 5 | 6 |
| --- | --- | --- | --- | --- | --- |
| *N*(*n*) для непрерывной функции | 2047 | 65535 | 2097151 | 67108863 | 2,15E+09 |
| *N*(*n*) липшицевой функции, =0.5 | 127 | 3412 | 102984 | 3215632 | 1,02E+08 |
| *N*(*n*) метод редукции, =0.75 | 862 | 14371 | 273379 | 5499292 | 1,14E+08 |
| *N*(*n*) метод редукции, =0.6 | 80 | 388 | 2163 | 12818 | 78622 |

Из таблицы видно, что даже при очень благоприятных условиях, связанных либо со свойствами функции, либо эффективными алгоритмическими решениями, когда  равно 0,5, сложность задачи глобальной оптимизации остается экспоненциальной. Этот факт позволяет разрабатывать приемлемые по сложности алгоритмы распараллеливания задачи глобальной оптимизации для сравнительно небольших размерностей. Метод редукции также принципиально не изменяет экспоненциальный характер роста сложности задачи глобальной оптимизации. Однако, при даже незначительном уменьшении коэффициента «прореживания»  существенно расширяется диапазон размерности задачи глобальной оптимизации, когда за счет распараллеливания можно добиться ощутимых конечных результатов.

Разработка параллельных алгоритмов глобальной оптимизации

Как мы отмечали ранее, в общем случае, методы оптимизации относятся к классу нерешаемых задач. В этом смысле, уже невозможно надеяться на создание универсального алгоритма оптимизации в равной степени эффективно решающего любую задачу. Выход из сложившейся ситуации можно найти на пути разработки частных алгоритмов оптимизации, учитывающих особенности математических моделей объектов исследования и специфику постановки задачи, а также применение технологий параллельных вычислений. Но для полномасштабного применения этого подхода необходимы специальные программные средства, позволяющие оперативно разрабатывать эффективные частные алгоритмы оптимизации. Более того, частные алгоритмы оптимизации должны опираться на современную парадигму параллельных алгоритмов.

В данном случае большое значение приобретают средства автоматизации разработки моделей и кодов параллельных алгоритмов, тестирования и сопровождения программных приложений (в частности алгоритмов оптимизации). Но поскольку технология программирования (особенно в параллельном исполнении) должна быть понятна конечному пользователю, специалисту в предметной области, то на первое место выступают визуальные средства автоматизации программирования параллельных приложений.

Методология разработки параллельных алгоритмов в корне отличается от традиционных методов создания последовательных кодов программ. В настоящее время более распространёнными и доступными являются вычислительных систем с распределенной памятью, в которых для организации информационного взаимодействия между процессорами в большинстве случаев используется интерфейс передачи данных MPI. Необходимость профессионального владения технологиями программирования на языке высокого уровня (C++, C#) и MPI еще дальше отдаляет «конечных» пользователей – специалистов в предметных областях, потребителей суперкомпьютерных технологий, от возможности участия в разработке параллельных алгоритмов. Одним из выходов из сложившейся ситуации является применение средств автоматизации проектирования и разработки параллельных программ с использованием выразительных визуальных, графических моделей описания алгоритмов.

Для неспециалиста MPI не знакомого с устройством кластера, в инструменте разработки параллельных программ в первую очередь необходимо иметь механизм автоматизированного управления памятью и синхронизацией. С другой стороны применение графической технологии программирования сокращает время на модификацию не только параллельных алгоритмом, но и последовательных разработок. Дополнительные модули тестирования и проверки алгоритмов необходимы для предотвращения типичных ошибок параллельного программирования, предварительной оценки эффективности разрабатываемого алгоритма и оптимизации под конкретную аппаратную платформу вычислительной системы. Средствам и способам разработки параллельных алгоритмов посвящена следующая глава.