# Применение алгоритмов машинного обучения для повышения эффективности алгоритма многокритериальной оптимизации

Баркалов К.А., Козинов Е.А.

# 1. Введение

# 2. Постановка задачи

In the most general form, the multi-objective optimization (MMO) problem can be formulated as follows

|  |  |
| --- | --- |
|  | (1) |

где есть objective functions (критерии эффективности), есть вектор варьируемых параметров, а есть размерность решаемой задачи multi-objective оптимизации. Множество возможных значений параметров (search domain) обычно представляет собой *N*-мерный гиперкуб

|  |  |
| --- | --- |
|  | (2) |

при заданных граничных векторах *a* и *b*.

Не уменьшая общности, предполагается, что objective functions должны быть минимизированы для повышения показателей эффективности принимаемых решений *.* При этомпредполагается, что , *,* являются многоэкстремальными представлены в виде time-consuming “black-box” computational procedures. Предполагается также, что objective functions , *,* удовлетворяют условию Липщица

|  |  |
| --- | --- |
|  | (3) |

where , are the Lipschitz constants and denotes the Euclidean norm in . Условие (3) означает, что при небольших вариациях параметра *y*∈*D* соответствующие изменения значений functions , *,* являются ограниченными.

# 3. Метод решения

В рамках предлагаемого подхода решение задач MMO выполняется в несколько этапов. В рамках первого этапа задача MMO сводится к решению серии скалярных задач глобальной оптимизации. На втором этапе выполняется редукция размерности. На последнем этапе осуществляется совместное решении серии задач глобального поиска. Далее рассматривается общая схема скаляризации задач MMO, используемый метод понижения размерности, а также метод решения порождаемых задач глобального поиска.

* 1. **Scalarization of multiple objective functions**

В самом общем виде, задача глобальной оптимизации, порождаемая при скаляризации multiple objective functions задачи МMО, может быть представлена в виде

|  |  |
| --- | --- |
| , , | (4) |

где есть скалярная многоэкстремальная функция, получаемая в результате скаляризации objective functions , *,* а есть вектор параметров используемой свертки functions. В литературе приводятся много методов построения функции : минимаксная свертка [XXX], метод оптимизации по образцу [XXX], метод -ограничений [XXX] и т.д. В выполненных вычислительных экспериментах раздела 5 без уменьшения общности предлагаемого подхода использовалась минимаксная свертка частных критериев

|  |  |
| --- | --- |
| , , | (5) |

где , , функции оптимизируемых критериев из (1) для decision , а коэффициенты есть показатели значимости каждого критерия. Обычно можно предполагать, что область возможных значений коэффициентов представляется собой множество

|  |  |
| --- | --- |
| *s*. | (6) |

Следует отметить, что в силу (3) скалярная функция из (5) также удовлетворяет условию Липщица c некоторой константой *L*, т.е.

|  |  |
| --- | --- |
|  | (7) |

Отметим, что критерии задачи (1) как правило являются противоречивыми. Скалятизация критериев (4) позволяет найти лишь один из возможных вариантов решения. Для обоснованного выбора оптимального варианта, как правило, необходимо решить целое семейство задач (4)

|  |  |
| --- | --- |
|  | (8) |

Необходимость поиска решения целого семейства задач (8) существенно повышает сложность решаемой задачи.

* 1. **Dimensionality reduction**

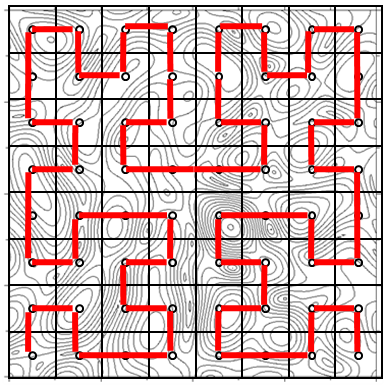
Задачи семейства (8) содержат несколько оптимизируемых параметров. Задачи подобного вида являются вычислительно сложными и подвержены «проклятию размерности» - вычислительная сложность возрастает экспоненциально при увеличении числа параметров [XXX]. Для решения подобных задач могут применяться различные подходы как строящие неравномерные покрытия многомерной обрасти [XXX], так и методы позволяющие снизить размерность решаемой задачи [XXX].

В разработанном подходе используется редукция размерности на основе *кривых* или *разверток* Пеано *y*(*x*), однозначно и непрерывно отображающих отрезок [0,1] на *N*-мерный гиперкуб - см., например, [XXX]. В результате такой редукции многомерные задачи глобальной оптимизации (4) сводятся к одномерным задачам

|  |  |
| --- | --- |
|  | (9) |

Принцип уменьшения размерности представлен на рис. 1.

*y*(*x*)



*x*

0

1

Рис. 1. Пример использования разверток Пеано.

В начале, согласно используемому алгоритму, выбирается . После определения точки вычисляется многомерный образ с использованием отображения . В многомерной точке вычисляется значение исходной многомерной функции из (4). Вычисленное значение z =используется как значение редуцированной одномерной функции из (9)*.*

* 1. **Solving the one-dimensional reduced optimization problem**.

Отметим, что полученные критерии (9) удовлетворяют условию Гёльдера, т.е.

|  |  |
| --- | --- |
|  | (10) |

где константа определяется соотношением , есть константа Липшица из (7), а размерность задачи оптимизации (1). Наличие условия (10) позволяет использовать эффективные одномерные алгоритмы глобального поиска строящие неравномерные покрытия области поиска [XXX].

В рамках предлагаемого подхода решения задач MMO (1) используется информационно-статистическая теория глобального поиска [XXX]. Данная теория послужила основой для разработки большого количества эффективных методов многоэкстремальной оптимизации – см., например, [XXX наши статьи с Гергелем]. Приведем кратко общую вычислительную схему.

# 4. Методы повышения эффективности многокритериального поиска

Использование базового подхода к решению задач MMO предложенного в разделе 3 позволяет решать некоторые из поставленных задач, однако, эффективность подхода может быть значительно улучшена. Для повышения эффективности алгоритма поиска в рамках предлагаемого подхода применяется повторное использование данных в множестве поисковой информации (см. раздел 4.1), а также метод штрафа построенного на основе алгоритмов машинного обучения (см. раздел 4.2).

* 1. **Поэтапное решение множества задач глобальной оптимизации**.
  2. **Метод построения штрафной функции на основе алгоритмов машинного обучения**.

|  |  |
| --- | --- |
|  |  |

Рис.2. Примеры построенных разделяющих гиперплоскостей в пространстве оптимизируемых критериев

# 5. Результаты вычислительных экспериментов

Вычислительные эксперименты проводились на суперкомпьютере «Лобачевский» Нижегородского государственного университета (операционная система – CentOS 7, система управления – SLURM). Один узел суперкомпьютера располагает 2-я процессорами Intel Sandy Bridge E5-2660 2.2 GHz, 64 Gb RAM. Центральный процессор является 8-и ядерным (т.е. всего на узле доступно 16 ядер CPU). Для получения исполняемого программного кода использовался компилятор Intel C++ 17.0.0, python 3.9 and scikit-learn 0.24.2. The numerical experiments were performed using the Globalizer system [XXX].

Первая серия экспериментов была выполнена для сравнения алгоритма MGSA с рядом широко известных алгоритмов многокритериальной оптимизации на примере решения тестовой двухкритериальной задачи [XXX]

|  |  |
| --- | --- |
|  | (35) |

В ходе экспериментов для решения задачи (35) выполнялось построение численной аппроксимации области Парето. Качество аппроксимации оценивалось с помощью показателей the hypervolume and distribution uniformity indexes (HV). [37, 40]. Первый из этих показателей характеризует полноту аппроксимации (большее значение соответствует более полному покрытию области Парето), а второй – равномерность покрытия (меньшее значение соответствует более равномерному покрытию области Парето).

В рамках данного эксперимента сравнивались пять алгоритмов многокритериальной оптимизации: the Monte-Carlo (MC) method, the genetic algorithm SEMO from the PISA library [42], the Non-uniform coverage (NUC) method [40], the bi-objective Lipschitz optimization (BLO) method [39] и алгоритм MGSA, предложенного в данной статье. For the first three algorithms, the numerical results were used from [41]. The results of the BLO method were presented in [39].

Для MGSA было решено 25 задач (4) при разных значениях коэффициентов свертки , равномерно распределенных в Λ. При проведении экспериментов параметр надежности был задан , а параметр точности . В полном виде результаты выполненных экспериментов представлены в таблице 1.

Таблица 1. Сравнение эффективности алгоритмов многокритериальной оптимизации

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Метод решения** | MC | SEMO | NUC | BLO | **MGSA**  **()** | **MGSA**  **()** |
| **Количество  итераций метода** | 500 | 500 | 515 | 498 | **302** | **269** |
| **Количество найденных точек  Парето границы** | 67 | 104 | 29 | 68 | **92** | **79** |
| **HV индекс** | 0.300 | 0.312 | 0.306 | 0.308 | **0.312** | **0.312** |
| **DU индекс** | 1.277 | 1.116 | 0.210 | 0.175 | **0.101** | **0.103** |

Как показывают результаты выполненных экспериментов, использование базового алгоритма MGSA (т.е. при ) дает заметное преимущество по сравнению с рассмотренными методами многокритериальной оптимизации даже при решении сравнительно простых задач МКО. Использование методов машинного обучения позволяет дополнительно сократить число выполняемых испытаний глобального поиска.

Во второй серии экспериментов проводилось решение двухкритериальных двумерных задач MMO, т.е. , . В качестве критериев задачи использовались многоэкстремальные функции, определяемых соотношениями [XXX]:

|  |  |
| --- | --- |
|  | (36) |

where

,

are defined in the range , and the parameters are the independent random numbers distributed uniformly. На рис. 3 показаны линии уровней нескольких функций из данного семейства.

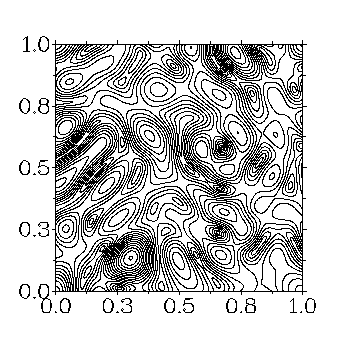
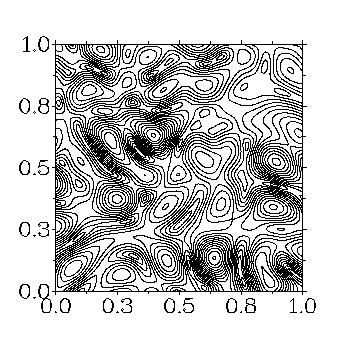
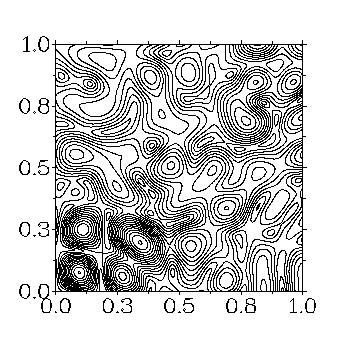


Fig. 3. Линии уровня нескольких многоэкстремальных функций   
используемых во второй серии вычислительных экспериментов

На начальном этапе решения серии задач MMO необходимо было выбрать значение параметра из (XXX). Выбранное значение показателя из (XXX) должно обеспечивать приемлемое качество решения задач с точки зрения показателей HV и DU, а также сокращение количество испытаний решения серии задач. Для более обоснованного выбора значения параметра из (XXX) рассматривалась одна задача из серии. Параметры точности , надежности и количества решаемых скалярных задач (4) равное 50 были зафиксированы. Значения параметра из (XXX) менялось в диапазоне [0..0.1]. Изменение показателей HV и DU, а также количество испытаний в зависимости от параметра показаны на рис. 4.

|  |  |
| --- | --- |
|  |  |

Fig. 4. Изменение показателей построенной области Парето в одной из задач MMO в зависимости от выбранного показателя   
(слева – изменение числа итераций, справа – изменение показателей HV и DU)

Изменение заполнения области поиска при использовании алгоритмов машинного обучения показано на рис. 5.

|  |  |
| --- | --- |
|  |  |

Fig. 5. Точки испытаний в зависимости от выбранного значения параметра

Исходя из рисунка 2 видно, что увеличение параметра из (XXX) приводит к существенному уменьшению числа испытаний, однако, вместе с тем с увеличением значения параметра из (XXX) ухудшаются значения параметров HV и DU. Исходя из проведенной серии экспериментов было зафиксировано значение . На рисунке 5 видно, что использование значения параметра позволяет уменьшить число испытаний вне области Парето, вместе с тем в окрестности области Парето плотность точек испытаний остается высокой.

Для построения более обоснованных выводов об эффективности разработанного подхода было выполнено решение 100 многокритериальных задач, формируемых с использованием многоэкстремальных функций семейства (36). В таблице 2 приведены усредненные результаты вычислительных экспериментов.

Таблица 2. Сравнение эффективности алгоритмов многокритериальной оптимизации  
с использованием и без использования алгоритмов машинного обучения   
при решении серии из 100 задач MMO

|  |  |  |
| --- | --- | --- |
|  | MGSA (α=0.0) | MGSA (α=0.01) |
| Average number of iterations | 1902,4 | 658,6 |
| Average value of  DU indexe (less is better) | 1,32 | 1,55 |
| Average value of  HV indexe (more is better) | 92,1 | 91,9 |
| Reducing the number of iterations | 1 | 2,9 |

Исходя из таблицы 2 видно, что за счет применения алгоритмов машинного обучения удалось добиться существенного сокращения количества испытаний (почти в 3 раза) для решения серии задач MMO с сохранением схожих значений показателей HV и DU.

# Литература