APPLICATION OF MACHINE LEARNING METHODS FOR CROSS-CLASSIFICATION OF ALGORITHMS AND PROBLEMS OF MULTIVARIATE CONTINUOUS OPTIMIZATION

A. Chepurnov[1], N. Ershov[1,2,a]

[1] *Moscow State University, Faculty of Computational Mathematics and Cybernetics; Russia, 119991, Moscow, GSP-1, 1-52, Leninskiye Gory;*

[2] *Dubna State University, Institute of the system analysis and management; 141980, Russia, Moscow Region, Dubna, Universitetskaya str., 19;*

E-mail: [a] *ershovnm@gmail.com*

The paper is devoted to the development of a software system for the mutual classification of families of population optimization algorithms and problems of multivariate continuous optimization. One of the goals of this study is to develop methods for predicting the performance of the algorithms included in the system and choosing the most effective algorithms from them for solving a user-specified optimization problem. In addition, the proposed software system can be used to expand existing test suites with new optimization problems. The work was carried out with the financial support of the Russian Foundation for Basic Research (Grant No. 20-07-01053 A).

Keywords: optimization methods, genetic algorithms, swarm optimization, multivariate data analysis.

Chepurnov Andrey, Ershov Nikolay

ПРИМЕНЕНИЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ ДЛЯ КРОСС-КЛАССИФИКАЦИИ АЛГОРИТМОВ И ЗАДАЧ МНОГОМЕРНОЙ НЕПРЕРЫВНОЙ ОПТИМИЗАЦИИ

А.В Чепурнов[1], Н.М. Ершов[1,2,a]

[1] *Факультет вычислительной математики и кибернетики, МГУ им. М. В. Ломоносова; 119991, Москва, ГСП-1, Ленинские горы, д. 1, стр. 52, факультет ВМК.*

[2] *Институт системного анализа и управления, ГБОУ ВО МО «Университет «Дубна»; 141980, Московская область, г. Дубна, ул. Университетская, 19.*

E-mail: [a] ershovnm@gmail.com

Предлагаемая работа посвящена разработке программной системы для проведения взаимной классификации семейств популяционных алгоритмов оптимизации и задач многомерной непрерывной оптимизации. Одной из целей настоящего исследования является разработка методов предсказания эффективности работы включенных в систему алгоритмов и выбора из них наиболее эффективных алгоритмов для решения заданной пользователем задачи оптимизации. Кроме того, предлагаемая программная система может быть использована для расширения существующих тестовых наборов новыми задачами оптимизации. Работа выполнена при финансовой поддержке РФФИ (грант № 20-07-01053 А).

Ключевые слова: методы оптимизации, генетические алгоритмы, роевая оптимизация, многомерный анализ данных

Андрей Чепурнов, Николай Ершов

1. Введение

За последние десятилетия было создано множество алгоритмов, которые используются в решении широкого круга оптимизационных задач. В частности, продолжают разрабатываться новые алгоритмы оптимизации «черного ящика» (Black-Box Optimization), которые не требуют никакой дополнительной информации о решаемой задаче, кроме целевой функции и области значений параметров.

Хотя многие из этих алгоритмов демонстрируют отличные результаты применительно к задачам малой и средней размерности, крупноразмерные задачи с несколькими сотнями или тысячами параметров всё ещё очень трудны для оптимизации [1]. В связи с этим, отдельный интерес представляет возможность проведения сравнительного анализа эффективности работы алгоритмов непрерывной оптимизации для задач большой размерности. Однако информация о сходимости решения на отдельно взятых задачах не позволяет сделать выводы об эффективности предлагаемых алгоритмов. Возникает необходимость в специализированных инструментах сравнительного анализа.

2. Система тестирования

Поскольку поведение алгоритмов многомерной оптимизации может сильно зависеть от ландшафта целевой функции оптимизации, для получения наиболее полной информации об эффективности работы исследуемых алгоритмов их требуется протестировать на наиболее полном наборе тестовых задач. Заметим, что каждый такой алгоритм с фиксированным набором гиперпараметров можно представить как точку $(x_1,x_2,\ldots,x_n)$ в некотором многомерном пространстве алгоритмов, где x_i — некоторые характеристики его работы — например, k-тое вычисленное значение целевой функции для некоторой задачи. Тогда сравнение эффективности работы алгоритмов сводится к анализу точек в n-мерном пространстве — к применению методов автоматической классификации и снижения размерности. Аналогично можно обозначить задачу сравнения тестовых задач, основываясь на результатах работы алгоритмов над ними.

В существующих наборах тестовых задач в качестве тестовых функций содержатся комбинаций аналитических функций из известного «классического» списка тестовых функций непрерывной оптимизации, которые составляются таким образом, чтобы полученная целевая функция обладала некоторыми необходимыми свойствами, влияющими на сложность ландшафта. В список этих функций входит функция Растригина, функция Розенброка, функция Экли и другие. В качестве примеров можно привести набор, используемый при проведении конкурса методов непрерывной оптимизации на ежегодном конгрессе IEEE по эволюционным вычислениям (IEEE CEC) [2], бенчмарк COCO (COmparing Continuous Optimizers) [3], пакеты тестовых функций в составе MATLAB [4].

Однако такой подход к тестированию имеет некоторые недостатки. Результат тестирования алгоритма на подобной задаче — только график значений целевой функции и время работы алгоритма. Этого может быть недостаточно, чтобы делать выводы при сравнении эффективности работы алгоритмов в пространстве большой размерности, но для аналитически заданной тестовой функции сложно получить какую-либо дополнительную информацию. Поэтому, помимо тестовых функций, мы решили также рассматривать содержательные модельные задачи, решения для которых можно дополнительно интерпретировать и делать выводы о их качестве — например, использовать не только значение целевой функции, но и значение погрешности относительно точного решения. В качестве таких задач на данный момент мы используем задачу аппроксимации полиномами и задачу укладки графа на плоскости.

Стоит отметить, что проведение сравнительного анализа обширных классов алгоритмов на обширных классах задач требует больших вычислительных затрат и невозможно без использования вычислительных кластеров. В связи с этим мы использовали собственную

систему тестирования алгоритмов с применением технологий MPI и OpenMP, осуществляющую автоматический перебор параметров тестовых задач и гиперпараметров алгоритма и сбор статистики.

На данный момент в системе реализованы следующие алгоритмы (описаны в [5]):

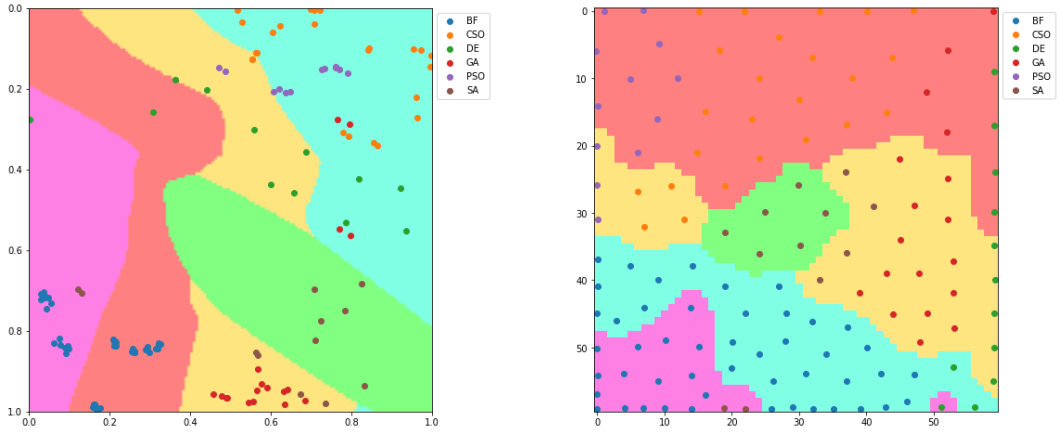
- Particle Swarm Optimization (PSO) – метод роя частиц;
- A Competitive Swarm Optimizer (CSO) – метод конкурирующего роя;
- Bacterial Foraging optimization (BF) – алгоритм бактериального поиска;
- Genetic Algorithm (GA) – генетический алгоритм;
- Differential evolution (DE) – алгоритм дифференциальной эволюции;
- Simulated annealing (SA) – алгоритм имитации отжига.

С учётом различных вариаций операторов и значений гиперпараметров количество версий алгоритмов в каждом классе достигает нескольких десятков. Всего было протестировано 130 алгоритмов, для каждого из которых производилось по 5 запусков. Всего было произведено 7700 тестовых запусков, 10000 вычислений целевой функции в каждом. Ручная обработка такого объёма данных затруднительна и требует автоматических средств анализа.

3. Методы анализа

В качестве алгоритма кластеризации использовался алгоритм k-средних. Начиная с 4 кластеров значение функции кластеризации слабо изменяется, поэтому было выбрано разбиение на 5 кластеров.

Для понижения размерности были испробованы несколько методов, первый из них — автокодировщик со скрытым слоем из 2 нейронов [6]. С его помощью возможно визуализировать полученные данные, в том числе найденные кластеры, подавая на вход обученного декодировщика координаты точек изображения и применяя к результатам алгоритм k-средних (рис. 1а). Преимуществам использования автокодировщиков является то, что архитектура таких сетей позволяет регулировать сложность получаемого преобразования. Эксперименты показали, что одного дополнительного слоя в кодировщике и декодировщике достаточно, чтобы визуализация получалась достаточно наглядной и не была излишне сложной.



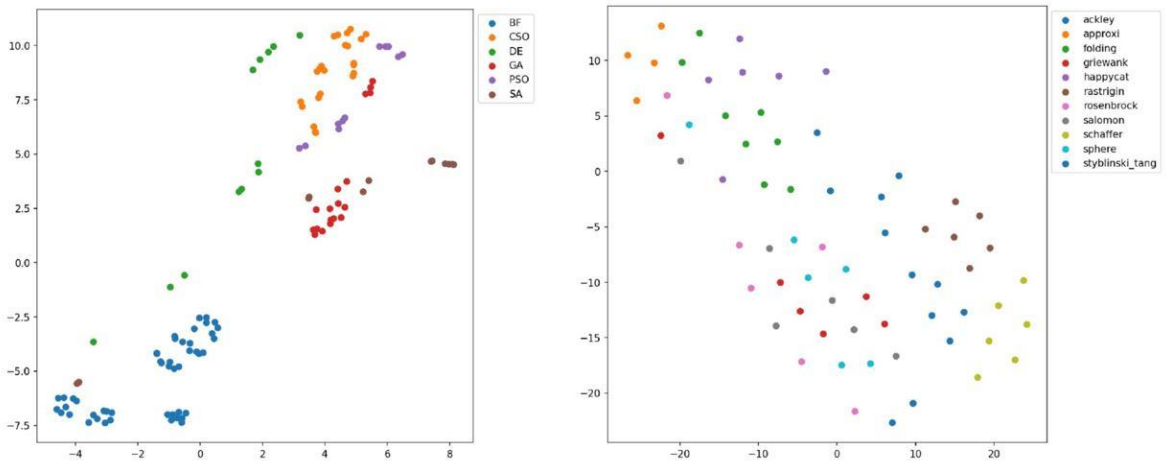
а) б)

Рисунок 1. Кластеризация алгоритмов методом k-средних и понижение размерности обученным автокодировщиком (а) и самоорганизующейся картой Кохонена (б).

Вторым был рассмотрен метод самоорганизующихся карт Кохонена [7], который тоже позволяет визуализировать кластеры. Помимо прочего, метод позволяет регулировать кучность

точек изменением параметров функции соседства. Наиболее удачной оказалась конфигурация, при которой точки располагаются на плоскости достаточно равномерно (рис. 1б), поскольку иначе кластеры трудноразличимы.

Третий рассмотренный метод — «Стохастическое вложение соседей с t-распределением» (tSNE) [8]. В отличии от предыдущих методов, нелинейная проекция tSNE сама по себе хорошо выделяет кластеры алгоритмов (рис. 2а). Примечательно, что и алгоритм k-средних, и tSNE выделили классы генетических алгоритмов, имитации отжига, алгоритмов бактериального поиска с и без сигнальной коммуникации между бактериями. Метод роя частиц, метод конкурирующего роя и алгоритм дифференциальной эволюции оба метода выделяют в отдельный кластер (tSNE менее явно).



а) б)

Рисунок 2. Понижение размерности алгоритмом tSNE пространства алгортмов (а) и задач (б).

С классификацией задач рассмотренные методы справляются значительно хуже. Алгоритм k-средних выделяет кластеры несистематично, задачи из одного класса оказываются в разных кластерах. tSNE проецирует некоторые классы кучно (например, задачу укладки графов), но не выделяет явных кластеров и тоже не справляется с задачей (рис. 2б). Скорее всего это связано с тем, что размерность пространства задач больше, а самих задач пока недостаточно много.

4. Направления дальнейшей работы

Основным этапом нашей дальнейшей работы будет разработка методов предсказания эффективности работы алгоритмов. Уже сейчас очевидно, что это потребует проведения вычислений для целевых функций большей размерности и большего лимита на количество их вычислений. Также в систему будут включаться другие алгоритмы оптимизации. Как уже было отмечено выше, полнота набора тестовых задач влияет на качество анализа, поэтому его тоже планируется расширять.

Многие операторы стохастических алгоритмов не фиксируют тип оптимизации (непрерывная, дискретная, смешанная и т.д.), поэтому мы рассматриваем возможность добавления в систему задач комбинаторной оптимизации и методы сравнительного анализа для них.

Литература

[1] IEEE CEC'2021 Special Session and Competition on Large-Scale Global Optimization // TF LSGO URL: https://www.tflsgo.org/special_sessions/cec2021.html (дата обращения: 09.07.2021).

[2] Li X. et al. Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization //gene. – 2013. – Т. 7. – №. 33. – С. 8.

[3] Hansen N. et al. COCO: A platform for comparing continuous optimizers in a black-box setting //Optimization Methods and Software. – 2021. – Т. 36. – №. 1. – С. 114-144.

[4] Test functions for global optimization algorithms // File Exchange - MATLAB Central URL: https://www.mathworks.com/matlabcentral/fileexchange/23147-test-functions-for-global-optimization-algorithms (дата обращения: 09.07.2021).

[5] Карпенко А. П. Современные алгоритмы поисковой оптимизации. М.: Изд-во МГТУ им. Н.Э. Баумана, 2014

[6] Kramer M. A. Nonlinear principal component analysis using autoassociative neural networks //AIChE journal. – 1991. – Т. 37. – №. 2. – С. 233-243.

[7] Kohonen T. Self-organized formation of topologically correct feature maps //Biological cybernetics. – 1982. – Т. 43. – №. 1. – С. 59-69.

[8] Van der Maaten L., Hinton G. Visualizing data using t-SNE //Journal of machine learning research. – 2008. – Т. 9. – №. 11.