УДК 004.021

Эффективность использования генетических алгоритмов в решении задач оптимизации

Волков Иван Максимович
Российский экономический университет им. Г.В. Плеханова
Студент

Токмакова Наталия Романовна
Российский экономический университет им. Г.В. Плеханова
Студент

Аннотация
Генетические алгоритмы разработаны для решения задач поиска, получив широкое применение в задачах оптимизации. Такие задачи имеют целевую функцию, экстремум которой необходимо определить. Если функция имеет неклассический вид, то для поиска экстремума, скорее всего не выработаны конкретные алгоритмы. В данном случае можно использовать генетические алгоритмы, для реализации которых необходимо выработать функцию приспособленности и генетические операторы. В то же время такой подход имеет свои недостатки, например, ранняя сходимость, что приводит к возможному нахождению локального экстремума.
Ключевые слова: Генетический алгоритм, Оптимизация, Эффективность алгоритма

The efficiency of using genetic algorithms in solving optimization problems

Volkov Ivan Maksimovich
Plekhanov Russian University of Economics
Student

Tokmakova Natalia Romanovna
Plekhanov Russian University of Economics
Student

Abstract
Genetic algorithms are used to solve search problems, being widely used in optimization problems. Such tasks have an objective function, the extremum of which is necessary to determine. If the function has a non-classical form, then specific algorithms of finding the extremum have not likely been developed. In this case, you can use genetic algorithms, for the implementation of which it is necessary to develop a fitness function and genetic operators. But this approach has its drawbacks, for example, early convergence, which possibly leads to the finding of a local extremum.
Keywords: Genetic Algorithm, Optimization, Algorithm’s Efficiency

Необходимость в оптимизации алгоритмов поиска в рамках машинного обучения привела к разработке генетических алгоритмов, которые моделируют процессы случайного подбора, комбинирования и вариации искомых параметров. Механизмы, представленные в алгоритме подобны механизмам естественного отбора, существующие в природе, например, мутации, наследование, отбор и другие [1].

В теории генетических алгоритмов используются термины, взятые из биологии, такие как, популяция – это конечное множество особей. Особи представляют собой хромосомы – упорядоченные последовательности генов. Так, хромосомы могут представлять собой массив из нулей и единиц, которые являются генами. Генотип – это набор хромосом одной особи. Отсюда следует, что особь может состоять из одной и более хромосом.

Создание генетических алгоритмов было основано на принципах естественной эволюции, разработанных Ч. Дарвином и Г. Мендалем. Основы теории генетических алгоритмов сформулированы Дж. Г.Холландом [2] и в дальнейшем были развиты рядом других исследователей. Вопросы практического применения таких алгоритмов подробно рассматривал Д. Голдберг.

Упомянутые принципы представляют собой:

1. В 1859 году Дарвин сформировал концепцию выживания сильнейших в книге «Происхождение видов путем естественного отбора». В соответствии с данной концепцией особи, которые способны решать поставленные задачи лучше остальных, чаще выживают и размножаются. В генетическом алгоритме за определение лучших особей отвечает функция приспособленности, а за размножение – оператор скрещивания.
2. В 1865 году Мендель установил, что хромосома потомка состоит из хромосом его родителей. За это в алгоритме также отвечает оператор скрещивания.
3. Концепция мутации описывает такие свойства потомков, которые отсутствуют у родителей. Оператор мутации производит такие изменения с заданной небольшой вероятностью в алгоритме.

Использование генетических алгоритмов актуально в экономическом секторе. Такой алгоритм можно использовать, например, с целью построения стратегического плана развития компании. Качество выработанного плана критически важно, потому что в атмосфере конкурентной борьбы у каждой стороны есть своя стратегия. План должен предусматривать наибольший выигрыш при минимальных потерях. Числовые показатели будущего плана можно закодировать в хромосомах особей, а взаимосвязи таких показателей выразить в целевой функции.

Классический генетический алгоритм

Основная идея алгоритма заключается в том, что необходимо в цикле определять уровень приспособленности текущей популяции относительно некоторых целевых значений. [3] Если популяция соответствует необходимому глобальному экстремуму целевой функции, то она становится

результирующей и алгоритм завершается. В ином случае, производятся фазы скрещивания и мутации, в которых на основе текущего поколения формируется новое, содержащее комбинацию генов родителей, которая может быть, как более, так и менее близка к целевой функции. Это определяется с помощью *функции приспособленности*. Она возвращает значение для каждой особи популяции, с помощью которого можно определить близость текущего решения к целевой функции. На основе таких значений производится селекция и формируется новое поколение, в идеале более приближенное к целевой функции, чем предыдущее.

Блок-схема выше описанного алгоритма с критерием остановки «нахождение поколения, решающее поставленную задачу», выглядит следующим образом:

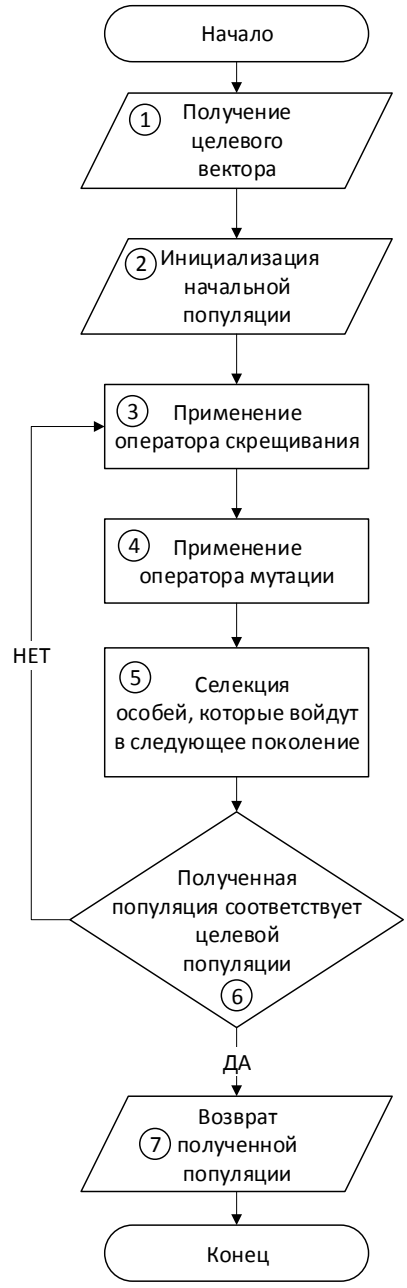

Рисунок 1 – Блок-схема классического генетического алгоритма

Другими критериями остановки алгоритма (шаг 7) могут являться «исчерпание отведенного числа поколений» или «исчерпание времени, отведенного на выполнение алгоритма».

На шаге 1 алгоритм получает целевой вектор, который представляет собой генотип, к которому результирующая популяция должна стремиться. Иными словами, вектор характеризует *целевую функцию*, речь о которой шла выше. В большинстве случаев генотип состоит из генов, представленных в виде отдельных битов, чисел или иных объектов в зависимости от интерфейса, предоставленного конкретной программой.

На шаге 2 случайным образом задается популяция. При генерации не стоит цели получения особей, отвечающим целевой функции, но сгенерированная популяция должна отвечать условиям соответствия интерфейсу особи, заданному в текущей реализации. Условие соответствия интерфейсу также включает в себя возможность расчета функции приспособленности для каждой отдельной особи.

На шаге 3 производится *скрещивание* (размножение). [4] Для этого выбирается несколько родителей (обычно два) по одному из следующих принципов:

1. Панмиксия: родители выбираются случайным образом. У каждой особи равные шансы.
2. Инбридинг: первый родитель выбирается случайно, остальные выбираются наиболее похожими на него.
3. Аутбридинг: первый родитель выбирается случайно, остальные выбираются наименее похожими на него.

Сама реализация скрещивания зависит от поставленной задачи и формата представления данных. Основная цель размножения – это предоставление возможности потомку унаследовать данные родителей в соответствии с определённым алгоритмом, который может быть как случайным, так и построенным по конкретным правилам. Согласно классическому генетическому алгоритму, скрещивание при двух родителях происходит следующим образом [5]:

1. Случайным образом выбирается точка скрещивания l_m. Предположим, что хромосома особи представляет собой массив, состоящий из m генов, тогда l_m принадлежит интервалу [0; m - 1].
2. Первый потомок формируется из генов, стоящих на индексах [0; l_m] первого родителя, и генов, стоящих на индексах [l_m + 1; m - 1] второго родителя;
3. Второй потомок формируется наоборот: из генов, стоящих на индексах [0; l_m] второго родителя, и из генов, стоящих на индексах [l_m + 1; m - 1] первого родителя.

Желательно, чтобы в размножении участвовали все особи текущей популяции, а не только наиболее приближенные к значениям целевой функции. Это необходимо для поддержания разнообразия в особях. Например, если некий генотип станет преобладающим среди наилучших особей (станет локальным максимумом), то вся популяция унаследуют

данный генотип, а особи станут копиями преобладающей особи. Использование всех особей ведет к увеличение вычислительной сложности алгоритма. Возможной оптимизацией может являться использование в размножении всех самых приспособленных и несколько менее приспособленных в условиях полной случайности выбора родителей. При таком подходе возрастает роль мутации.

На шаге 4 осуществляется мутация. Как и в природе, так и в алгоритме, вероятность ($p_{mutation}$) того, что произойдет мутация некого гена, мала. В классическом генетическом алгоритме такая вероятность обычно находится в промежутке [0; 0,1]. Необходимо отметить, что в то же время вероятность скрещивания ($p_{crossover}$) находится в промежутке [0,5; 1].

Одним из возможных способов реализации мутации может быть следующий алгоритм:

1. Генерируется значение $p_{mutation}$ случайным образом в промежутке [0; 0,1].
2. Хромосома представляет собой массив, состоящий из значений 0 и 1. Такие значения являются генами. Для каждого гена особи случайным образом генерируется значение из интервала [0; 1]
3. Если это значение совпадает со значением $p_{mutation}$, то такой ген меняет значение на противоположное (0 на 1 или 1 на 0).

На шаге 5 производится селекция среди всех имеющихся на текущий момент особей. Селекция производится по принципу естественного отбора, согласно которому наибольшие шансы на отбор имеют хромосомы с наиболее подходящими значениями, полученными из функции приспособленности.

Эффективность приведенного алгоритма зависит от следующих показателей:

1. Мощность популяции (N) – количество особей. Чем больше N, тем больше разнообразие потенциальных решений. Но с другой стороны, это повышает вычислительную сложность алгоритма. Чем меньше N, тем быстрее работает алгоритм, но высока вероятность схождения к локальному экстремуму. Обычно принимают, что $N \in [30; 200]$. Так, на разных этапах алгоритма могут быть различные значения N. Например, на первых этапах значение N может быть большим и уменьшаться на каждой итерации.
2. Вероятность скрещивания и Вероятность мутации. Значения данных вероятностей влияют на способность сходиться к оптимуму и способность находить новые области в пространстве решений.

Преимущества и недостатки использования генетических алгоритмов в задачах оптимизации

Генетический алгоритм может быть использован при решении любой проблемы, которая сформулирована как задача оптимизации [6]. Такие задачи требуют разработки структуры данных для представления целевого

решения, показателя качества решения и генетические операторы для порождения новых решений из старых.

Возможность настраивать внутренние алгоритмы операторов генетического алгоритма позволяет охватывать широкую область задач. Так, регулируя работу функции приспособленности или операторов мутации, скрещения или других, можно сузить область применения конкретного алгоритма, а также комбинировать разработанный алгоритм с другими, более традиционными, алгоритмами, применимыми к текущей задаче.

Немаловажно отметить, что эволюционные процессы в природе происходят параллельно. Генетический алгоритм, который естественным образом эмулирует эволюционные процессы, может быть реализован таким же образом: тяжелые для вычисления функции приспособленности могут вычисляться параллельно, а этап селекции производиться последовательно. Так, можно решать более сложные задачи. Генетический алгоритм естественно реализуется в многопроцессорных распределенных системах.

Самое важное, что генетические алгоритмы могут применяться в исследовании таких проблем, решений и конкретных алгоритмов, которых еще не существует. Таким образом, нивелируется проблемы, связанные с экспертными оценками и мнениями, которые в определенных случаях могут быть неквалифицированными, ошибочными или зависимыми от внешних обстоятельств.

В то же время, конфигурация генетического алгоритма, а особенно функции приспособленности, неочевидна. При постановки сложной задачи, для которой еще не был выработан алгоритм, необходимо спроектировать кодирование решения. Сюда входит определение функции приспособленности, выбор мощности популяции, вероятностей и иных показателей. Более того, сложно определить критерии останова алгоритма.

С точки зрения использования генетических алгоритмов для решения задач оптимизации, такие алгоритмы не подходят для работы с функциями с одним экстремумом (гладкими унимодальными функциями), для поиска локальных экстремумов. При решении мультимодальных задач возможна сходимость к локальным экстремумам, а, следовательно, не гарантируется нахождение глобального экстремума. Именно поэтому часто генетические операторы применяют на всю популяцию, а не только на те особи, у которых значение функции приспособляемости наилучшее. Но именно из-за такого подхода повышается вычислительная сложность алгоритма.

Согласно теореме No Free Lunch [7] невозможно выбрать такие генетически операторы и их параметры так, чтобы алгоритм давал лучшие результаты вне зависимости от поставленной задачи. Данная теорема была доказана в 1996 году Вольпертом и Макреди следующим образом. Пусть $P\left(d_m^y \mid f,\ m, a\right)$ – условная вероятность получения частного решения d_m после m итераций работы алгоритма a при целевой функции f. Тогда для любой пары алгоритмов a_1 и a_2 верно следующее равенство $P\left(d_m^y \mid f,\ m,\ a_1\right) = P\left(d_m^y \mid f,\ m,\ a_2\right)$. Следовательно, сумма условных вероятностей

посещения в пространстве решений каждой точки d_m одинакова для различных целевых функций независимо от алгоритма. Установим, что сложность (производительность) алгоритма $\Phi(d_m^y)$ в среднем для различных целевых функций f вероятность $P(\Phi(d_m^y) \mid f, m, a)$ не зависит от алгоритма a. Другими словами, не существует лучшего алгоритма для решения всех поставленных проблем. Если алгоритм хорошо решает конкретную задачу, то он не сможет решать другие задачи с той же эффективностью.

Оценка эффективности использования генетических алгоритмов

Для того, чтобы генетический алгоритм работал лучше, чем случайный поиск, необходимо отразить структуру поставленной проблемы в генетических операторах. Классические методы оптимизации достаточно эффективны и более широко применяются при поиске решения линейных, квадратичных, унимодальных, строго выпуклых и других специальных классов задач, которые хорошо исследованы. Генетические алгоритмы эффективны в другой области оптимизационных задач, где целевая функция:

1. имеет разрывы;
2. не дифференцируемая;
3. мультимодальная;
4. зашумленная.

Другими словами, генетические алгоритмы эффективны там, где целевые функции имеют нестандартный вид, что характерно при решении реальных практических задач.

Для оценки эффективности алгоритма необходимо доказать его сходимость и дать оценку вычислительной сложности, но в большинстве случаев это возможно при упрощенной постановке задачи. В таких случаях часто используют метод проверки алгоритмов на тестовых задачах, относящихся к рассматриваемой проблемной области, которые также необходимо разработать и/или подобрать готовые.

Таким образом, несмотря на то, что генетические алгоритмы просты и интуитивны в понимании, их сложно применять для нетипичных задач, относящихся к реальным проблемам по причине неочевидного построения целевой функции и генетических операторов. Более того, полученный результат сложно проверить, так как такой алгоритм имеет высокую вероятность сходимости на локальных экстремумах.

Библиографический список

1. Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы. М: ФИЗМАТЛИТ, 2006.
2. John H. Holland Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. - MA, USA: MIT Press Cambridge, 1992.
3. Darrell Whitley A Genetic Algorithm Tutorial // Satistics and Computing. 1994.

№4 (2).
4. Lothar M. Schmitt Theory of genetic algorithms // Theoretical Computer Science. 2001.
5. Классический генетический алгоритм. Часть IV. Скрещивание, мутация, создание популяции // Портал искусственного интеллекта URL: http://www.aiportal.ru/articles/genetic-algorithms/classic-alg-part4.html (дата обращения: 18.12.2018).
6. Преимущества и недостатки генетических алгоритмов. // НОУ Интуит URL: https://www.intuit.ru/studies/courses/14227/1284/lecture/24168 (дата обращения: 17.12.2018).
7. Macready W.G., Wolpert D. H. No Free Lunch Theorems for Search Technical. // Report SFI-TR-95-02-010. Sania Fe Instituie. 1996.