ISSN 2306-1561
Automation and Control in Technical Systems (ACTS)
2014, No 4, pp. 82-99.
DOI: 10.12731/2306-1561-2014-4-9

Genetic Algorithms in Problems of Rational Organization of Information Processes

Le Quang Hieu

Socialist Republic of Vietnam, Undergraduate Student, Department of «Automated Control Systems».

State Technical University – MADI, 125319, Russian Federation, Moscow, Leningradsky prospekt, 64. Tel.: +7 (499) 151-64-12. http://www.madi.ru

hieu_lee_hb@mail.ru

Surkova Nataliya Evgenievna

Russian Federation, Ph. D., Associate Professor, Department of «Automated Control Systems».

State Technical University – MADI, 125319, Russian Federation, Moscow, Leningradsky prospekt, 64. Tel.: +7 (499) 151-64-12. http://www.madi.ru

sneee@mail.ru

Ostroukh Andrey Vladimirovich

Russian Federation, full member RAE, Doctor of Technical Sciences, Professor, Department of «Automated Control Systems».

State Technical University – MADI, 125319, Russian Federation, Moscow, Leningradsky prospekt, 64. Tel.: +7 (499) 151-64-12. http://www.madi.ru

ostroukh@mail.ru

Abstract. Discusses the use of genetic algorithms to optimize the rational organization of information processes in the system of special-purpose distributed data processing. It is shown that for the high demands of stability and efficiency of the control characteristic of some special systems necessary to carry out optimization of the operation of such systems. To find the optimal solutions of the corresponding optimization problems is proposed the use of genetic algorithms.

Keywords: genetic algorithm, search query, relevancy, filtering, ranking, population, crossover, mutation, selection, fitness.

ISSN 2306-1561
Автоматизация и управление в технических системах (АУТС)
2014. – №4. – С. 82-99.
DOI: 10.12731/2306-1561-2014-4-9

УДК 004.9

Генетические алгоритмы в задачах рациональной организации информационно-вычислительных процессов

Ле Куанг Хиеу

Социалисти́ческая Респу́блика Вьетна́м, магистрант кафедры «Автоматизированные системы управления».

ФГБОУ ВПО «Московский автомобильно-дорожный государственный технический университет (МАДИ)», 125319, Российская Федерация, г. Москва, Ленинградский проспект, д.64, Тел.: +7 (499) 151-64-12, http://www.madi.ru

hieu_lee_hb@mail.ru

Суркова Наталия Евгеньевна

Российская Федерация, кандидат педагогических наук, доцент кафедры «Автоматизированные системы управления».

ФГБОУ ВПО «Московский автомобильно-дорожный государственный технический университет (МАДИ)», 125319, Российская Федерация, г. Москва, Ленинградский проспект, д.64, Тел.: +7 (499) 151-64-12, http://www.madi.ru

sneee@mail.ru

Остроух Андрей Владимирович

Российская Федерация, академик РАЕ, доктор технических наук, профессор кафедры «Автоматизированные системы управления».

ФГБОУ ВПО «Московский автомобильно-дорожный государственный технический университет (МАДИ)», 125319, Российская Федерация, г. Москва, Ленинградский проспект, д.64, Тел.: +7 (499) 151-64-12, http://www.madi.ru

ostroukh@mail.ru

Аннотация. Рассматриваются вопросы использования генетических алгоритмов оптимизации для рациональной организации информационно-вычислительных процессов в системах специального назначения с распределённой обработкой данных. Показано, что для обеспечения высоких требований устойчивости и оперативности управления, характерных для некоторых специальных систем, необходимо проведение оптимизации функционирования таких систем. Для поиска оптимальных решений

соответствующих оптимизационных задач предлагается использование генетических алгоритмов.

Ключевые слова: генетический алгоритм, поисковый запрос, релевантность, фильтрация, ранжирование, популяция, скрещивание, мутация, селекция, приспособленность.

1. Введение

Генетические алгоритмы (ГА) – адаптивные методы поиска, которые используются для решения задач функциональной оптимизации. Представляют собой своего рода модели машинного исследования поискового пространства, построенные на эволюционной метафоре [1, 2, 3]. Характерные особенности: использование строк фиксированной длины для представления генетической информации, работа с популяцией строк, использование генетических операторов для формирования будущих поколений [4 – 23].

Генетические алгоритмы оперируют совокупностью особей (популяцией), которые представляют собой строки, кодирующие одно из решений задачи. Этим метод отличается от большинства других алгоритмов оптимизации, которые оперируют лишь с одним решением, улучшая его.

Генетические алгоритмы применяются для решения следующих задач:

- оптимизация функций;
- разнообразные задачи на графах (задача коммивояжера, раскраска и т.д.);
- настройка и обучение искусственной нейронной сети;
- задачи компоновки;
- составление расписаний;
- игровые стратегии;
- аппроксимация функций;
- искусственная жизнь;
- биоинформатика.

Преимущества генетических алгоритмов:

- универсальность;
- высокая обзорность поиска;
- нет ограничений на целевую функцию;
- любой способ задания функции.

Недостатки генетических алгоритмов:

- относительно высокая вычислительная стоимость;
- квазиоптимальность.

2. Схема функционирования генетического алгоритма

Из биологии известно, что любой организм может быть представлен своим фенотипом, который фактически определяет, чем является объект в реальном мире, и

генотипом, который содержит всю информацию об объекте на уровне хромосомного набора. При этом каждый ген, то есть элемент информации генотипа, имеет свое отражение в фенотипе. Таким образом, для решения задач необходимо представить каждый признак объекта в форме, подходящей для использования в генетическом алгоритме. Все дальнейшее функционирование механизмов генетического алгоритма производится на уровне генотипа, позволяя обойтись без информации о внутренней структуре объекта, что и обуславливает его широкое применение в самых разных задачах.

В наиболее часто встречающейся разновидности генетического алгоритма для представления генотипа объекта применяются битовые строки. При этом каждому атрибуту объекта в фенотипе соответствует один ген в генотипе объекта. Набор генов или один ген представляют собой закодированный признак.

Ген – это битовая строка, чаще всего фиксированной длины. Для кодирования гена в бинарной реализации генетического алгоритма часто используют код Грея (см. пример в таблице 1).

Таблица 1 - Пример фенотипа

Признак	**Двоичное значение признака**	**Десятичное значение признака**
Признак 1	0011	3
Признак 2	1100	12
Признак 3	1110	14
Признак 4	0111	7
Признак 5	1001	9

После определения фенотипа генетический алгоритм функционирует по следующей схеме действий (рисунок 1):

- Создание начальной популяции.
- Оценка особей популяции.
- Отбор (селекция).
- Скрещивание.
- Мутация.
- Формирование новой популяции.

Если популяция не сошлась, то 2, иначе – останов (прекращение функционирования генетического алгоритма).

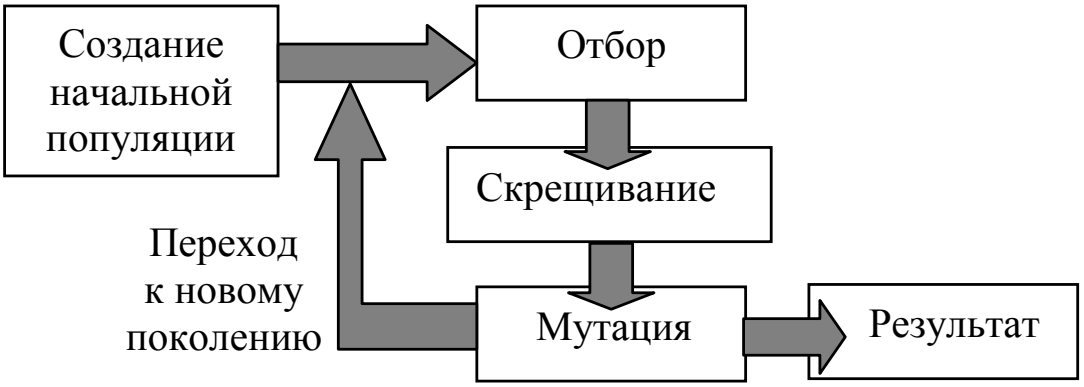

Рисунок 1 - Схема функционирования генетического алгоритма

Создание начальной популяции

Стандартный генетический алгоритм начинает свою работу с формирования начальной популяции – конечного набора допустимых решений задачи. Эти решения могут быть выбраны случайным образом или получены с помощью простых приближенных алгоритмов. Выбор начальной популяции не имеет значения для сходимости процесса, однако формирование «хорошей» начальной популяции (например, из множества локальных оптимумов) может заметно сократить время достижения глобального оптимума. Если отсутствуют предположения о местоположении глобального оптимума, то индивидов из начальной популяции желательно распределить равномерно по всему пространству поиска решения.

Популяция инициируется в начальный момент времени $t = 0$ и состоит из k особей, каждая из которых представляет возможное решение рассматриваемой проблемы. Особь – это одна или несколько хромосом (обычно одна). Хромосома состоит из генов, то есть это битовая строка (хромосомы не ограничены бинарным представлением, есть реализации генетического алгоритма, построенные на векторах вещественных чисел). Гены располагаются в различных позициях хромосомы и принимают значения, называемые аллелями.

Оценка особей популяции

Для решения задачи с помощью генетического алгоритма необходимо задать меру качества для каждого индивида в пространстве поиска. Для этой цели используется функция приспособленности (fitness function). Функция приспособленности должна принимать неотрицательные значения на ограниченной области определения, при этом совершенно не требуются непрерывность и дифференцируемость. Значение этой функции определяет, насколько хорошо подходит особь для решения задачи.

В задачах максимизации целевая функция часто сама выступает в качестве функции приспособленности, для задач минимизации целевую функцию следует инвертировать. Если решаемая задача имеет ограничения, выполнение которых невозможно контролировать алгоритмически, то функция приспособленности, как правило, включает также штрафы за невыполнение ограничений (они уменьшают ее значение).

Отбор (селекция)

На каждом шаге эволюции с помощью вероятностного оператора селекции (отбора) выбираются два решения-родителя для их последующего скрещивания. Среди операторов селекции наиболее распространенными являются два вероятностных оператора пропорциональной и турнирной селекции. В некоторых случаях также применяется отбор усечением.

Пропорциональный отбор (proportional selection)

Каждой особи назначается вероятность Ps(i), равная отношению ее приспособленности к суммарной приспособленности популяции. Затем происходит отбор (с замещением) всех n особей для дальнейшей генетической обработки согласно величине Ps(i).

Простейший пропорциональный отбор – рулетка – отбирает особей с помощью n «запусков» рулетки. Колесо рулетки содержит по одному сектору для каждого члена популяции. Размер i-го сектора пропорционален соответствующей величине Ps(i). При таком отборе члены популяции с более высокой приспособленностью с большей вероятностью будут чаще выбираться, чем особи с низкой приспособленностью.

Турнирный отбор

Турнирный отбор реализуется следующим образом: из популяции, содержащей m особей, выбирается случайным образом t особей, и наиболее приспособленная особь записывается в промежуточный массив (между выбранными особями проводится турнир). Эта операция повторяется m раз. Строки в полученном промежуточном массиве затем используются для скрещивания (случайным образом). Размер группы строк, отбираемых для турнира, часто равен 2. В этом случае говорят о двоичном/парном турнире.

Отбор усечением

Данная стратегия использует отсортированную по убыванию популяцию. Число особей для скрещивания выбирается в соответствии с порогом $T \in [0;1]$. Порог определяет, какая доля особей, начиная с самой первой (самой приспособленной), будет принимать участие в отборе. В принципе, порог можно задать и равным 1, тогда все особи текущей популяции будут допущены к отбору. Среди особей, допущенных к скрещиванию случайным образом m/2 раза, выбираются родительские пары, потомки которых образуют новую популяцию.

Ранговый отбор

Этот вид отбора подразумевает следующее: для каждой особи ее вероятность попасть в промежуточную популяцию пропорциональна ее порядковому номеру в отсортированной по возрастанию приспособленности популяции. Такой вид отбора не зависит от средней приспособленности популяции.

Элитный отбор

Элитные методы отбора гарантируют, что при отборе обязательно будут выживать лучший или лучшие члены популяции. Наиболее распространена процедура обязательного сохранения только одной лучшей особи, если она не прошла, как другие, через процесс отбора, кроссовера и мутации. Элитизм может быть внедрен практически в любой стандартный метод отбора. Использование элитизма позволяет не потерять хорошее промежуточное решение, но в то же время из-за этого алгоритм может «застрять» в локальном экстремуме. В большинстве случаев элитизм не вредит поиску решения, и главное – это предоставить алгоритму возможность анализировать разные хромосомы из пространства поиска.

Скрещивание

Как известно, в теории эволюции важную роль играет то, каким образом признаки родителей передаются потомкам. В генетических алгоритмах за передачу признаков родителей потомкам отвечает оператор, который называется оператором скрещивания (его также называют кроссовер или кроссинговер). Этот оператор определяет передачу признаков родителей потомкам, к ним применяется вероятностный оператор скрещивания, который строит на их основе новые (1 или 2) решения-потомки. Отобранные особи подвергаются кроссоверу (иногда называемому рекомбинацией) с заданной вероятностью Pc. Если каждая пара родителей порождает двух потомков, для воспроизводства популяции необходимо скрестить m/2 пары. Для каждой пары с вероятностью Pc применяется кроссовер. Следовательно, с вероятностью 1 - Pc кроссовер не происходит, и тогда неизмененные особи переходят на следующую стадию (мутации).

Существует большое количество разновидностей оператора скрещивания. Простейший одноточечный кроссовер работает следующим образом.

Сначала случайным образом выбирается одна из возможных точек разрыва. (Точка разрыва – участок между соседними битами в строке.) Обе родительские структуры разрываются на два сегмента по этой точке. Затем соответствующие сегменты различных родителей склеиваются и получаются два генотипа потомков:

Родитель 1: 1 0 0 1 0 1 1 | 0 1 0 0 1 **Потомок 1: 1 0 0 1 0 1 1 | 0 0 1 1 1**
Родитель 2: 0 1 0 0 0 1 1 | 0 0 1 1 1 **Потомок 2: 0 1 0 0 0 1 1 | 0 1 0 0 1**

В настоящее время исследователи ГА предлагают много других операторов скрещивания. Двухточечный кроссовер и равномерный кроссовер – вполне достойные альтернативы одноточечному оператору. В двухточечном кроссовере выбираются две точки разрыва, и родительские хромосомы обмениваются сегментом, который находится между двумя этими точками. В равномерном кроссовере каждый бит первого потомка случайным образом наследуется от одного из родителей; второму потомку достается бит другого родителя.

Мутация

Следующий генетический оператор предназначен для того, чтобы поддерживать разнообразие особей в популяции, – это оператор мутации. После того как закончится стадия кроссовера, потомки могут подвергаться случайным модификациям. В простейшем случае в каждой хромосоме, которая подвергается мутации, каждый бит с вероятностью Pm изменяется на противоположный (это так называемая одноточечная мутация):

Особь до мутации: **1 0 0 1 0 1 1 0 0 1 1 1**
Особь после мутации: **1 0 0 1 0 1 0 0 0 1 1 1**

Более сложной разновидностью мутации являются операторы инверсии и транслокации. Инверсия – это перестановка генов в обратном порядке внутри произвольно выбранного участка хромосомы:

Особь до инверсии: **1 0 0 1 1 1 1 0 0 1 1 1**
Особь после инверсии: **1 0 0 1 0 0 1 1 1 1 1 1**

Транслокация – это перенос какого-либо участка хромосомы в другой сегмент этой же хромосомы:

Особь до транслокации: **1 0 0 1 1 1 1 0 0 1 1 1**
Особь после транслокации: **1 1 1 0 0 0 1 1 0 1 1 1**

Все перечисленные генетические операторы (одноточечный и многоточечный кроссовер, одноточечная мутация, инверсия, транслокация) имеют схожие биологические аналоги.

Формирование нового поколения

После скрещивания и мутации особей необходимо решить проблему: какие из новых особей войдут в следующее поколение, а какие – нет, и что делать с их предками. Есть два наиболее распространенных способа.

1. Новые особи (потомки) занимают места своих родителей. После этого наступает следующий этап, в котором потомки оцениваются, отбираются, дают потомство и уступают место своим «детям».

2. Следующая популяция включает в себя как родителей, так и их потомков.

Во втором случае необходимо дополнительно определить, какие из особей родителей и потомков попадут в новое поколение. В простейшем случае в него после каждого скрещивания включаются две лучшие особи из четверки родителей и их потомков. Более эффективным является механизм вытеснения, который реализуется таким образом, что стремится удалять «похожие» хромосомы из популяции и оставлять отличающиеся.

Критерии останова

Другой важный момент функционирования алгоритма – определение критериев останова. Вообще говоря, такой процесс эволюции может продолжаться до бесконечности. Обычно в качестве критериев останова применяются или ограничение на максимальное число эпох функционирования алгоритма, или определение его сходимости, обычно путем сравнивания приспособленности популяции на нескольких эпохах и остановки при стабилизации этого параметра.

Схождением называется такое состояние популяции, когда все строки популяции почти одинаковы и находятся в области некоторого экстремума (рисунок 2).

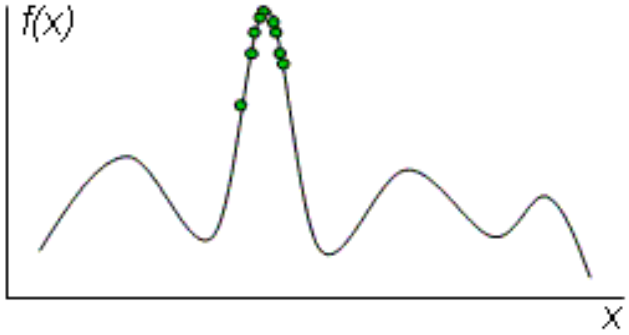

Рисунок 2 – Схождение генетического алгоритма

В такой ситуации кроссовер практически никак не изменяет популяции. А вышедшие из этой области за счет мутации особи склонны вымирать, так как чаще имеют меньшую приспособленность, особенно если данный экстремум является глобальным максимумом. Таким образом, схождение популяции обычно означает, что найдено лучшее или близкое к нему решение.

3. Модели генетических алгоритмов

Существуют различные модели генетического алгоритма (классический, простой генетический алгоритм, гибридный, CHC генетический алгоритм и др.) [1, 2, 3]. Они различаются по стратегиям отбора и формирования нового поколения особей, операторами генетического алгоритма, кодированием генов и т.д.

CHC-алгоритм

CHC (Cross generational elitist selection, Heterogenous recombination, Cataclysmic mutation) был предложен Эсхелманом и характеризуется следующими параметрами:

Для нового поколения выбираются N лучших различных особей среди родителей и детей. Дублирование строк не допускается.

Для скрещивания выбирается случайная пара, но не допускается, чтобы между родителями было мало хэммингово расстояние или мало расстояние между крайними различающимися битами.

Для скрещивания используется разновидность однородного кроссовера HUX (Half Uniform Crossover): ребенку переходит ровно половина битов каждого родителя.

Размер популяции небольшой, около 50 особей. Этим оправдано использование однородного кроссовера.

CHC противопоставляет агрессивный отбор агрессивному кроссоверу, однако все равно малый размер популяции быстро приводит ее к состоянию, когда создаются только более или менее одинаковые строки. В таком случае CHC применяет cataclysmic mutation: все строки, кроме самой приспособленной, подвергаются сильной мутации (изменяется около трети битов). Таким образом, алгоритм перезапускается и далее продолжает работу, применяя только кроссовер.

Genitor

Этот алгоритм был создан Д. Уитли. Genitor-подобные алгоритмы отличаются от классического ГА следующими тремя свойствами:

На каждом шаге только одна пара случайных родителей создает только одного ребенка.

Этот ребенок заменяет не родителя, а одну из худших особей популяции (в первоначальном Genitor – самую худшую).

Отбор особи для замены производится по ее рейтингу, а не по приспособленности.

В Genitor поиск гиперплоскостей происходит лучше, а сходимость быстрее, чем у классического генетического алгоритма, предложенного Холландом.

Гибридные алгоритмы

Идея гибридных алгоритмов (hybrid algorithms) заключается в сочетании генетического алгоритма с некоторым другим методом поиска, подходящим в данной задаче. В каждом поколении каждый полученный потомок оптимизируется этим методом, после чего производятся обычные для генетического алгоритма действия.

Такой вид развития называется ламарковой эволюцией, при которой особь способна обучаться, а затем полученные навыки записывать в собственный генотип, чтобы потом передать их потомкам. И хотя такой метод ухудшает способность алгоритма искать решение с помощью отбора гиперплоскостей, однако на практике гибридные алгоритмы оказываются очень удачными. Это связано с тем, что обычно велика вероятность того, что одна из особей попадет в область глобального максимума и после оптимизации окажется решением задачи.

Параллельные генетические алгоритмы

Генетические алгоритмы можно организовать как несколько параллельно выполняющихся процессов, это увеличит их производительность.

Рассмотрим переход от классического генетического алгоритма к параллельному. Для этого будем использовать турнирный отбор. Заведем N/2 процесса (здесь и далее процесс подразумевается как некоторая машина, процессор, который может работать независимо). Каждый из них будет выбирать случайно из популяции 4 особи, проводить 2 турнира и скрещивать победителей. Полученные дети будут записываться

в новое поколение. Таким образом, за один цикл работы одного процесса будет сменяться целое поколение.

Островная модель

Островная модель (island model, рисунок 3) – это тоже модель параллельного генетического алгоритма. Она заключается в следующем: пусть у нас есть 16 процессов и 1600 особей. Разобьем их на 16 подпопуляций по 100 особей. Каждая их них будет развиваться отдельно с помощью некого генетического алгоритма. Таким образом, можно сказать, что мы расселили особи по 16-ти изолированным островам.

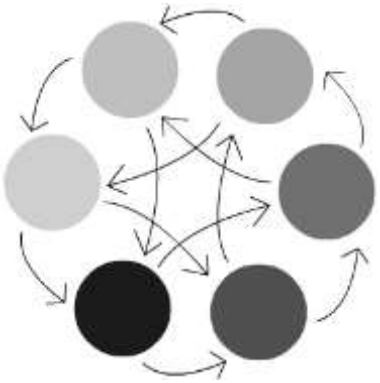

Рисунок 3 – Островная модель генетического алгоритма

Изредка (например, каждые 5 поколений) процессы (или острова) будут обмениваться несколькими хорошими особями. Этот процесс называется миграцией. Миграция позволяет островам обмениваться генетическим материалом.

Так как населенность островов обычно бывает невелика, подпопуляции будут склонны к преждевременной сходимости. Поэтому важно правильно установить частоту миграции. Чересчур частая миграция (или миграция слишком большого числа особей) приведет к смешению всех подпопуляций, и тогда островная модель будет несильно отличаться от обычного генетического алгоритма. Если же миграция будет слишком редкой, то она не сможет предотвратить преждевременного схождения подпопуляций.

Генетические алгоритмы стохастичны, поэтому при разных запусках популяция может сходиться к разным решениям (хотя все они в некоторой степени «хорошие»). Островная модель позволяет запустить алгоритм сразу несколько раз и пытаться совмещать «достижения» разных островов для получения в одной из подпопуляций наилучшего решения.

Ячеистые генетические алгоритмы

Ячеистые генетические алгоритмы (Cellular Genetic Algorithms) – модель параллельных генетических алгоритмов. Пусть дано 2500 процессов, расположенных на сетке размером 50×50 ячеек, замкнутой, как показано на рисунке 4 (левая сторона замыкается с правой, верхняя – с нижней, получается тор).

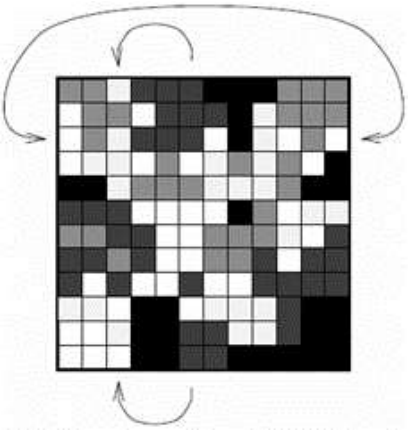

Рисунок 4 – Ячеистый генетический алгоритм

Каждый процесс может взаимодействовать только с четырьмя своими соседями (сверху, снизу, слева, справа). В каждой ячейке находится ровно одна особь. Каждый процесс будет выбирать лучшую особь среди своих соседей, скрещивать с ней особь из своей ячейки и одного полученного ребенка помещать в свою ячейку вместо родителя.

По мере работы такого алгоритма возникают эффекты, похожие на островную модель. Сначала все особи имеют случайную приспособленность (на рисунке она определяется по цвету). Спустя несколько поколений образуются небольшие области похожих особей с близкой приспособленностью. По мере работы алгоритма эти области растут и конкурируют между собой.

4. Пример работы и анализа генетического алгоритма

При использовании генетического алгоритма для решения задачи оптимизации необходимо:

1. Определить количество и тип оптимизируемых переменных задачи, которые необходимо закодировать в хромосоме.
2. Определить критерий оценки особей, задав функцию приспособленности (целевую функцию).
3. Выбрать способ кодирования и его параметры.
4. Определить параметры ГА (размер популяции, тип селекции, генетические операторы и их вероятности, величина разрыва поколений).

Отметим, что параметры ГА, определяемые в пункте 4 (а также, иногда, в пунктах 2 и 3), часто определяются методом проб и ошибок, на основе анализа получаемых результатов. Для анализа результатов работы ГА необходимо произвести несколько запусков алгоритма, для повышения достоверности выводов о качестве получаемых результатов, т.к. результат работы ГА носит вероятностный характер. Описанная общая схема решения задачи с использованием ГА показана на рисунке 5.

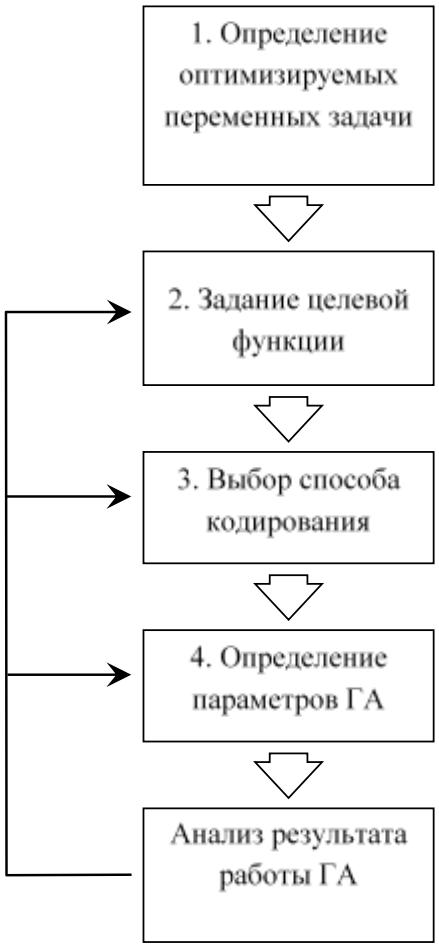

Рисунок 5 – Общая схема решения задачи с использованием ГА

Рассмотрим пример использования ГА для решения задачи минимизации следующей функции (сферическая функция):

$$z = \sum_{i=1}^{n} x_i^2, n = 10, x_i \in [-5{,}12;\ 5{,}11], \quad (1)$$

$$z \to \min .$$

Параметр n задает количество переменных функции z. Необходимо найти такие значения переменных x_i , при которых функция z принимает наименьшее значение. Будем использовать общую схему решения (рисунок 5):

1. Определение неизвестных переменных задачи. По условию поставленной задачи необходимо найти значения переменных xi, минимизирующие значение функции z, поэтому в хромосоме будем кодировать значения xi. Таким образом, каждый i-й ген хромосомы будет соответствовать i-й переменной функции z.

2. Задание функции приспособленности. Будем определять приспособленность особи в зависимости от значения, которое принимает функция z при подстановке в нее вектора параметров, соответствующих хромосоме этой особи. Поскольку рассматривается задача минимизации функции z, то будем также считать, что чем

меньше значение z, тем приспособленнее особь. Приспособленность i-й особи fi будем определять по следующей формуле:

$$f_i = z_i,$$

где zi – значение функции z в точке, соответствующей i-й особи.

3. Выбор способа кодирования. В качестве способа представления генетической информации рассмотрим целочисленное кодирование с точностью кодирования параметров 0,01. Тогда в имеющемся по условию задачи диапазоне изменения значений параметров [–5,12; 5,11] можно закодировать (5,12 – (-5,11))/0,01 + 1 = 1024 различных значений переменной. Единица прибавляется, так как значение переменной равное 0 также учитывается.

Для того чтобы представить 1024 различных значений переменной, достаточно использовать $\log_2 1024 = 10$ бит на каждую переменную. Таким образом, будет использоваться целочисленное кодирование с 10-разрядными генами.

4. Определение параметров ГА. Для решения задачи рассмотрим популяцию из 20 особей. При отборе особей для скрещивания будем использовать турнирную селекцию с бинарным турниром. В качестве генетических операторов будем использовать 1-точечный кроссинговер и битовую мутацию. Вероятности применения операторов скрещивания и мутации установим равными 0,7 и 0,05, соответственно. Новое поколение будем формировать только из особей-потомков, т.е. величина разрыва поколений T равна 1.

Результат работы генетического алгоритма с выбранными параметрами представлен на рисунке 6.

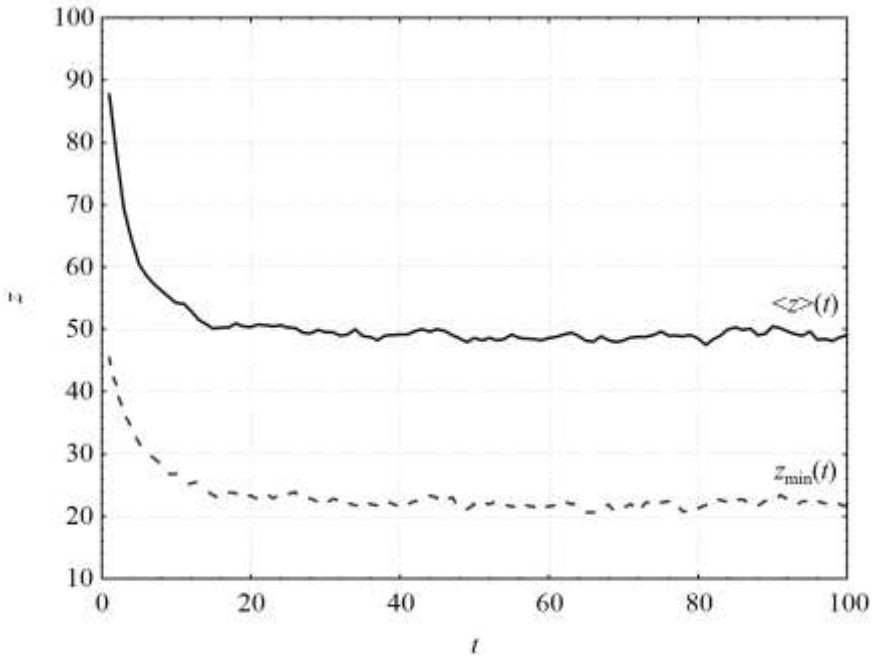

Рисунок 6 – Изменение zmin(t) и <z>(t). Популяция из 20 особей, бинарный турнирный отбор, одноточечный кроссинговер (PC = 0,7), битовая мутация (PM = 0,05)

Показаны зависимости изменения среднего <z> и наименьшего z_{min} в популяции значения функции z от номера поколения t. Данные усреднены по 100 независимым запускам.

Из рисунка 6 видно, что после 20-го поколения значение zmin колеблется в достаточно большом диапазоне. Из этого следует, что потери хороших особей в результате мутации велики, и следует уменьшить вероятность мутации. Установим значение этого параметра равным L-1 = 0,01, где L – длина хромосомы в битах, в данном случае L = 100. Результаты работы ГА с измененным значением вероятности мутации показаны на рисунке 7.

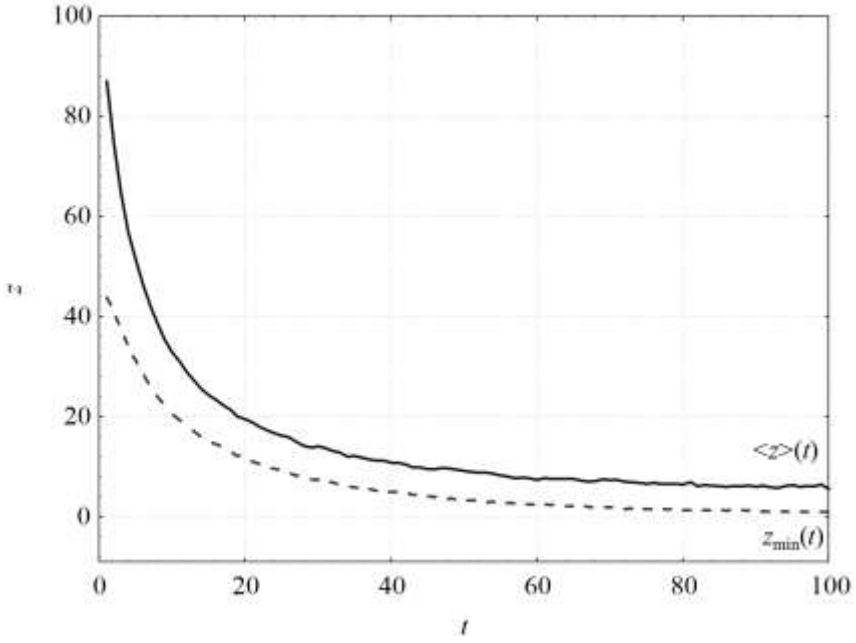

Рисунок 7 – Изменение zmin(t) и <z>(t). Популяция из 20 особей, бинарный турнирный отбор, одноточечный кроссинговер (PC = 0,7), битовая мутация (PM = 0,01)

Из сравнения графиков на рисунках 6 и 7 следует, что уменьшение вероятности мутации улучшило результат работы ГА. Также отметим, что теперь эволюционный процесс стабилизировался значительно позднее, примерно после 60-го поколения. Усредненное по всем запускам минимальное значение функции z, достигнутое за первые 100 поколений, равно ~1,016. Чтобы улучшить результат, увеличим давление селекции путем увеличения размера турнира до 4. Результат представлен на рисунке 8.

Увеличение давления селекции привело к ускорению эволюционного поиска за счет удаления из популяции особей со средней и плохой приспособленностью. В результате стабилизация наступила после 40-го поколения, а усредненное по всем запускам минимальное полученное значение функции z равно ~0,013. Наименьшее значение функции z достигается в точке xi = 0, i = 1,2,…,10 и равно 0. В случае поиска минимума функции z с точностью 0,01, для ГА с параметрами, соответствующими

графикам на рисунке 8, решение было найдено в 69 запусках из 100. При этом в среднем было использовано 1698,68 вычислений целевой функции.

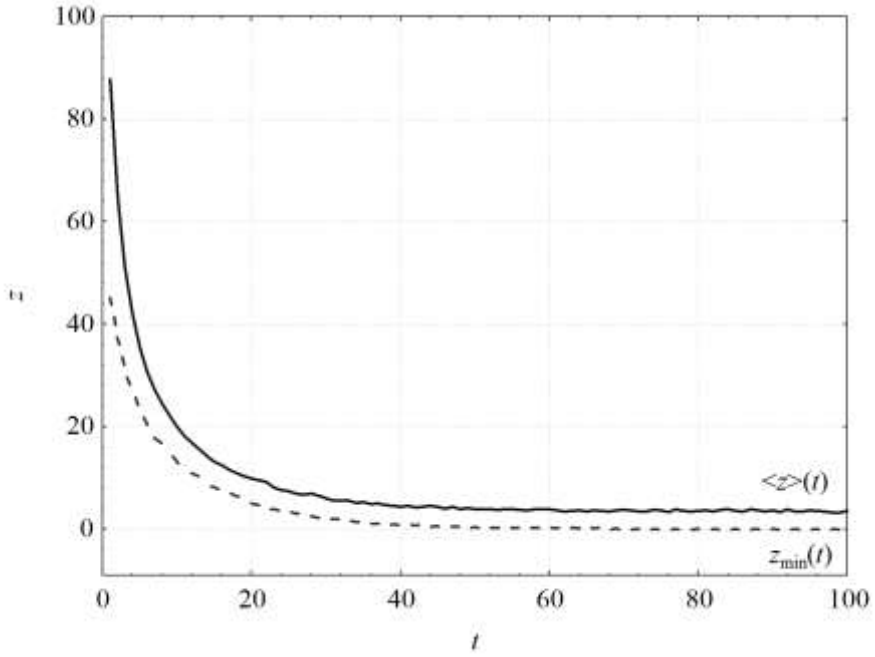

Рисунок 8 – Изменение zmin(t) и <z>(t). Популяция из 20 особей, турнирный отбор (t = 4), одноточечный кроссинговер (PC = 0,7), битовая мутация (PM = 0,01)

Чтобы повысить стабильность результатов, увеличим размер популяции до 50 особей. Полученные кривые zmin(t) и <z>(t) изображены на рисунке 9.

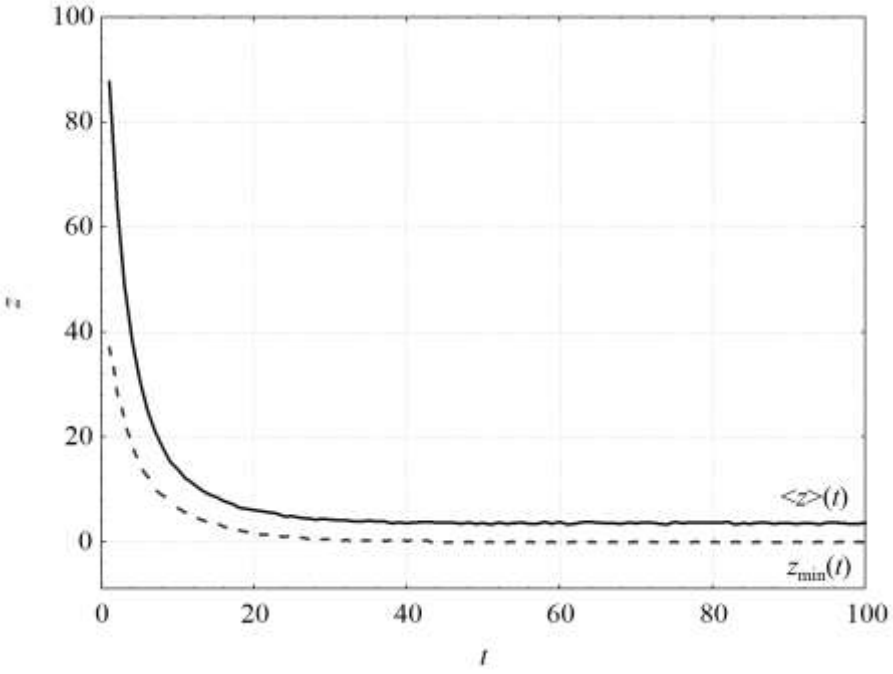

Рисунок 9 – Изменение zmin(t) и <z>(t). Популяция из 50 особей, турнирный отбор (t = 4), одноточечный кроссинговер (PC = 0,7), битовая мутация (PM = 0,01)

Во всех 100 запусках найден минимум функции z с точностью не меньше 0,01. Среднее количество вычислений целевой функции, использованное для нахождения решения, равно 3145,34.

5. Заключение

Генетические алгоритмы являются универсальным методом оптимизации многопараметрических функций, что позволяет решать широкий спектр задач.

Генетические алгоритмы имеют множество модификаций и сильно зависят от параметров. Зачастую небольшое изменение одного из них может привести к неожиданному улучшению результата.

Следует помнить, что применение генетических алгоритмов полезно лишь в тех случаях, когда для данной задачи нет подходящего специального алгоритма решения.

Список информационных источников

[1] Остроух А.В. Основы построения систем искусственного интеллекта для промышленных и строительных предприятий: монография / А.В. Остроух. – М.: ООО «Техполиграфцентр», 2008. – 280 с. – ISBN 978-5-94385-033-2.

[2] Остроух А.В. Системы искусственного интеллекта в промышленности, робототехнике и транспортном комплексе: монография / А.В. Остроух – Красноярск: Научно-инновационный центр, 2013. – 326 с. – ISBN 978-5-906314-10-9.

[3] Остроух А.В. Информационные технологии в научной и производственной деятельности / [ред. А.В. Остроух] – М: ООО "Техполиграфцентр", 2011. – 240 с. – ISBN 978-5-94385-056-1.

[4] Остроух А.В. Ввод и обработка цифровой информации: учебник для нач. проф. образования / А.В. Остроух. – М.: Издательский центр «Академия», 2012. – 288 с. – ISBN 978-5-7695-9457-1.

[5] Николаев А.Б. Информационные технологии в менеджменте и транспортной логистике: учебное пособие / А.Б. Николаев, А.В. Остроух. – Saint-Louis, MO, USA: Publishing House Science and Innovation Center, 2013. – 254 с. – ISBN 978-0-615-67110-9.

[6] Остроух А.В. Основы информационных технологий: учебник для сред. проф. образования / А.В. Остроух. – М.: Издательский центр «Академия», 2014. – 208 с. – ISBN 978-5-4468-0588-4.

[7] Васюгова С.А. Исследование перспектив и проблем интеграции человека с компьютером: искусственный интеллект, робототехника, технологическая сингулярность и виртуальная реальность / С.А. Васюгова, А.В. Остроух, М.Н. Краснянский, А. Самаратунга // Перспективы науки. – Тамбов: «ТМБПринт», 2011. – № 4(19). – С. 109-112.

[8] Белоусова А.И. Подход к формированию многоуровневой модели мультиагентной системы с использованием миваров / А.И. Белоусова, О.О. Варламов, М.Н. Краснянский, А.В. Остроух // Перспективы науки. – Тамбов: «ТМБПринт», 2011. – № 5(20). – С. 57-61.

[9] Варламов О.О. Анализ возможностей миварного подхода для систем искусственного интеллекта и современной робототехники / О.О. Варламов, А.В.

Остроух, М.Н. Краснянский, Т.Л. Давыдова // Вестник ТГТУ. – 2011. – Т.17. – № 3. – С.687-694.

[10] Суркова Н.Е. Методы проектирования информационных систем / Н.Е. Суркова, А.В. Остроух – М.: РосНОУ, 2004. – 144 с. – ISBN 5-89789-021-8.

[11] Суркова Н.Е. Методология структурного проектирования информационных систем: монография / Н.Е. Суркова, А.В. Остроух. – Красноярск: Научно-инновационный центр, 2014. – 190 с. – ISBN 978-5-906314-16-1.

[12] Николаев А.В. Использование словаря-справочника данных для реализации пользовательских средств обработки информации / А.В. Николаев, А.В. Остроух, С.А. Будихин, А.П. Баринов // Приборы и системы. Управление, контроль, диагностика. – М.: «Научтехлитиздат», 2008. – №3. – С. 13-15.

[13] Пшеничный Д.А. Анализ параметров и сравнение СУБД для реализации информационного обеспечения промышленного предприятия / Д.А. Пшеничный, А.В. Будихин, А.В. Остроух // Промышленные АСУ и контроллеры. – М.: «Научтехлитиздат», 2010. – №12. – С. 7-11.

[14] Помазанов А.В. Методика оптимизации баз данных / А.В. Помазанов, А.В. Остроух, А.И. Белоусова, А.О. Васильева // В мире научных открытий. Серия «Проблемы науки и образования». – 2012. – №12. – С.49-54.

[15] Помазанов А.В., Остроух А.В. Создание и тестирование распределённой системы работы с удалёнными узлами // Автоматизация и современные технологии. – 2014. – №7. – С. 17-23.

[16] Краснянский М.Н., Карпушкин С.В., Обухов А.Д., Молоткова Н.В., Галыгина И.В., Остроух А.В. Структура системы электронного документооборота для управления научно-образовательной деятельностью высшего учебного заведения // Промышленные АСУ и контроллеры. – 2014. – №8. – С. 23-31.

[17] Помазанов А.В., Остроух А.В. Новый подход к разработке прототипа распределенной системы баз данных промышленного предприятия // Промышленные АСУ и контроллеры. – 2014. – №9. – С. 11-20.

[18] A.V. Ostroukh, M.N. Krasnyanskiy, S.V. Karpushkin, A.D. Obukhov. Development of Automated Control System for University Research Projects // Middle East Journal of Scientific Research. 2014. Vol. 20 (12). pp. 1780-1784. DOI: 10.5829/idosi.mejsr.2014.20.12.21091.

[19] A. Ostroukh, A. Pomazanov. Realtime Development and Testing of Distributed Data Processing System for Industrial Company // Middle East Journal of Scientific Research. 2014. Vol. 20 (12). pp. 2184-2193. DOI: 10.5829/idosi.mejsr.2014.20.12.21106.

[20] Krasnyanskiy M.N., Karpushkin S.V., Obukhov A.D., Ostroukh A.V. Automated control system for university research projects // International Journal of Advanced Studies (iJAS). 2014. Vol. 4, Issue 1, pp. 22-26. DOI: 10.12731/2227-930X-2014-1-4.

[21] A. Ostroukh, V. Nikonov, I. Ivanova, T. Morozova, V. Strakhov. Distributed System of Real Time Head Gesture Recognition in Development of Contactless Interfaces // Middle East Journal of Scientific Research. 2014. Vol. 20 (12). pp. 2177-2183. DOI: 10.5829/idosi.mejsr.2014.20.12.21105.

[22] A. Ostroukh, V. Nikonov, I. Ivanova, T. Morozova, K. Sumkin, D. Akimov. Development of Contactless Integrated Interface of Complex Production Lines // Journal of Artificial Intelligence (JAI). 2014. Vol. 7, No 1. pp. 1-12. DOI: 10.3923/jai.2014.1.12.

[23] Morozova T., Sumkin K., Akimov D., Ostroukh A. Contactless integrated interface of production lines // International Journal of Advanced Studies (iJAS). 2014. Vol. 4, Issue 1, pp. 32-38. DOI: 10.12731/2227-930X-2014-1-6.