ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Московский институт электроники и математики им. А.Н. Тихонова

Шебаршин Павел Владимирович

АЛГОРИТМ ВСТРАИВАНИЯ ИНФОРМАЦИИ В ЦИФРОВЫЕ ИЗОБРАЖЕНИЯ С ПРИМЕНЕНИЕМ АНСАМБЛЯ МЕТАЭВРИСТИК

Выпускная квалификационная работа – магистерская диссертация
по направлению 10.04.01 «Информационная безопасность»
студента образовательной программы магистратуры
«Наименование образовательной программы»

Студент
П.В. Шебаршин

Научный руководитель
уч. степень, уч. звание
А.С. Мельман

Рецензент
д-р техн. наук, доцент кафедры
компьютерных систем в управлении и
проектировании
Д.В. Кручинин

Москва 2024

Аннотация

Методы встраивания информации в цифровые знаки, в частности, стеганография и цифровые водяные знаки доказали свою эффективность в обеспечении скрытой передачи данных и аутентификации цифровых изображений соответственно. Повышение требований к кибербезопасности требует повышения эффективности методов внедрения данных.

Метаэвристические алгоритмы оптимизации привлекают внимание как перспективный подход к улучшению качества встраивания. Хотя традиционные метаэвристики широко используются в этой области, потенциал их совместного использования является перспективным направлением для изучения.

В данной работе проводится сравнительное исследование трех подходов к построению ансамблей метаэвристик для поиска оптимальных вариантов встраивания в фазовый спектр дискретного преобразования Фурье (DFT). Данный алгоритм является алгоритмом безошибочного встраивания, он обеспечивает безошибочное извлечение внедренных данных и высокую незаметность встраивания благодаря метаэвристической оптимизации.

Данное исследование подчеркивает необходимость дальнейшего изучения современных алгоритмов оптимизации для улучшения методов встраивания данных. Работа содержит 56 страниц, 14 рисунков, 14 таблиц, 29 источников, 0 приложений.

Abstract

Methods of embedding information in digital signs, in particular, steganography and digital watermarks have proven their effectiveness in providing hidden data transmission and authentication of digital images, respectively. Increasing cybersecurity requirements requires improving the effectiveness of data injection methods.

Metaheuristic optimization algorithms attract attention as a promising approach to improving the quality of embedding. Although traditional metaheuristics are widely used in this field, the potential for their joint use is a promising area for study.

In this paper, a comparative study of three approaches to the construction of ensembles of metaheuristics is carried out to find optimal options for embedding the discrete Fourier transform (DFT) into the phase spectrum. This algorithm is an error-free embedding algorithm, it provides error-free extraction of embedded data and high invisibility of embedding due to metaheuristic optimization.

This study highlights the need for further study of modern optimization algorithms to improve data embedding methods.

СОДЕРЖАНИЕ

1 Введение

Современный мир требует постоянного совершенствования и оптимизации различных процессов. Одним из способов достижения этой цели является использование метаэвристических алгоритмов оптимизации. Метаэвристика — это область искусственного интеллекта, которая занимается разработкой алгоритмов, основанных на эвристических методах оптимизации. Она позволяет решать сложные задачи оптимизации, которые не могут быть решены классическими методами, либо же решение данной задачи путем прямого перебора является недопустимым по времени.

Стеганография сообщений в цифровых изображениях — это техника сокрытия сообщений в цифровых изображениях таким образом, что они незаметны для невооруженного глаза. Это достигается путем внесения незначительных изменений в пиксели изображения, которые не влияют на его воспринимаемое качество.

Оптимизация в стеганографии цифровых изображений важна по нескольким причинам. Во-первых, оптимизация позволяет скрывать больше данных в изображении без ущерба для его воспринимаемого качества. Это достигается путем использования более эффективных методов стеганографии и оптимизации параметров встраивания данных. Во-вторых, оптимизация помогает сделать сокрытые данные менее заметными для невооруженного глаза и автоматических систем обнаружения. Это достигается путем тщательного выбора пикселей, в которые встраиваются данные, и минимизации изменений в значениях пикселей. В-третьих, оптимизация повышает надежность сокрытых данных, гарантируя, что они могут быть успешно извлечены даже после сжатия, обработки или других модификаций изображения. Это достигается путем использования надежных методов встраивания данных и добавления механизмов коррекции ошибок. В-четвертых, оптимизация может сократить время, необходимое для встраивания и извлечения данных из изображений. Это достигается путем использования эффективных алгоритмов и оптимизации кода.

Метаэвристики представляют собой класс методов оптимизации, которые позволяют находить приближенные решения сложных задач оптимизации, когда точное решение недостижимо за разумное время. В контексте стеганографии, применение метаэвристик имеет ряд преимуществ. Метаэвристики могут быть адаптированы под конкретную задачу стеганографии и учитывать специфические требования к скрытой информации, к изображению и к контексту использования. Задача стеганографии требует нахождения оптимального способа встраивания информации в изображение таким образом, чтобы она оставалась незамеченной. Это сложная оптимизационная задача, где метаэвристики могут помочь исследовать пространство возможных решений и находить близкие к оптимальным варианты. Многие методы метаэвристик способны искать решения в глобальном пространстве оптимизации, что позволяет избежать застревания в локальных оптимумах. Это важно для стеганографии, где требуется найти скрытую вставку информации, которая остается незамеченной. Методы метаэвристик обычно работают быстро и могут давать хорошие результаты даже на больших объемах данных, что важно для задач стеганографии, где необходимо оперативно встраивать информацию.

Ансамбли метаэвристик могут быть лучше, чем отдельные метаэвристики в задачах оптимизации стеганографического встраивания информации в цифровые изображения так как они повышают разнообразие решений задачи имея возможность взять сильные стороны разных алгоритмов, имеют возможность гибкой настройки работы метаэвристик, снижают шанс того, что будет найдено плохое решение так как результаты работы нескольких метаэвристик будут объединены тем или иным способом. Например, ансамбль, объединяющий генетический алгоритм, алгоритм роя частиц и алгоритм муравьиной колонии, может эффективно оптимизировать параметры стеганографического встраивания, используя сильные стороны каждой метаэвристики для поиска высококачественных решений. В целом, ансамбли метаэвристик предоставляют более надежный и эффективный

подход к оптимизации стеганографического встраивания информации в цифровые изображения, чем использование отдельных метаэвристик.

Цель данной работы — Повышение незаметности и ёмкости встраивания дополнительной информации в цифровые изображения за счёт применения ансамбля алгоритмов метаэвристической оптимизации. Для достижения этой цели были поставлены следующие задачи:

1. Изучить основные принципы работы метаэвристических алгоритмов оптимизации.
2. Выбрать несколько алгоритмов метаэвристической оптимизации для исследования.
3. Реализовать выбранные алгоритмы.
4. Построение подхода к организации ансамбля метаэвристик.
5. Провести эксперименты на тестовых функциях.
6. Разработка алгоритма встраивания информации в частотную область дискретного преобразования Фурье цифровых изображений с применением ансамбля метаэвристик.
7. Проведение вычислительных экспериментов с разработанным алгоритмом, направленных на оценку эффективности алгоритма по критериям незаметности и ёмкости встраивания.
8. Сделать выводы о эффективности и применимости данного подхода по использованию метаэвристических алгоритмов оптимизации.

2 Теоретические сведения

Метаэвристические алгоритмы оптимизации применяются для решения сложных задач оптимизации, которые трудно или невозможно решить традиционными методами. Они находят приближенные решения, которые могут быть близки к оптимальным, но не гарантируют точность [1].

Эти алгоритмы широко используются в различных областях, таких как проектирование, инженерия, экономика, финансы и медицина, помогая оптимизировать процессы, сократить время и затраты на производство, улучшить качество продукции и повысить эффективность. Преимущества метаэвристических алгоритмов включают их способность работать с большим количеством переменных и ограничений, гибкость и способность находить решения в сложных условиях.

В данной работе выбрано несколько алгоритмов метаэвристической оптимизации, таких как Artificial Bee Colony (ABC), Biogeography-Based Optimization (BBO), Culture Algorithm (CA), Ant Colony Optimization Continuous (ACO), Harris Hawks Optimization (HHO) и Invasive Weed Optimization (IWO), Differential Evolution (DE), Forensic-Based Investigation Optimization (FBIO), Gradient-Based Optimizer (GBO), Particle Swarm Optimization (PSO). Также для использования ансамбля метаэвристик мной было выбрано три подхода для их реализации, а именно MetaheuristicEnsembleAlternation, MetaheuristicEnsembleAvg, MetaheuristicEnsembleLearn, которые далее будут рассмотрены более подробно.

2.1 Метаэвристики

Перед созданием самих ансамблей были реализованы и протестированны метаэвристики. Основные виды метаэвристик базируются на поведении животных (Bio Based), на принципах эволюции (Evolutionary Based), на подходах из жизни людей (Human Based), математические (Math Based), на поведении роя частиц или организмов (Swarm Based), на физических явлениях (Physics Based). В метэвристиках можно выделить

похожие этапы работы алгоритмов такие как: инициализации, мутация и обновление популяции.

В метаэвристических алгоритмах оптимизации инициализация представляет собой процесс задания начальных условий или начальной популяции для алгоритма [2]. Этот шаг играет важную роль, поскольку от правильного выбора начального состояния или начальной популяции зависит эффективность работы алгоритма. Инициализация в метаэвристических алгоритмах оптимизации может быть выполнена различными способами в зависимости от конкретного алгоритма. Например, в генетических алгоритмах инициализация может заключаться в случайном создании начальной популяции особей, которая будет подвергаться эволюционному процессу. В муравьиных алгоритмах инициализация может включать размещение муравьев в начальных узлах графа. Цель инициализации в метаэвристических алгоритмах оптимизации состоит в том, чтобы обеспечить разнообразие исходных решений или состояний, что помогает избежать застревания в локальных оптимумах и способствует лучшей сходимости к глобальному оптимуму. Правильно настроенная инициализация может значительно повысить эффективность работы метаэвристического алгоритма и ускорить процесс поиска оптимального решения.

Мутация в метаэвристических алгоритмах оптимизации представляет собой процесс случайного изменения решений или состояний в популяции или пространстве поиска. Этот шаг играет ключевую роль в разнообразии исследуемых решений, способствуя выходу из локальных оптимумов и обеспечивая более широкий поиск оптимального решения. Мутация обычно применяется в алгоритмах, основанных на эволюционных стратегиях, таких как генетические алгоритмы. В процессе мутации случайно выбранные компоненты решения или особи изменяются с некоторой вероятностью. Это помогает внести разнообразие в популяцию и поможет изучать новые регионы пространства поиска. Мутация может принимать различные формы

в зависимости от конкретного алгоритма. Например, в генетических алгоритмах мутация может включать случайное изменение генов особи, а в муравьиных алгоритмах мутация может происходить путем изменения феромонов на путях муравьев. В целом, мутация является важным компонентом метаэвристических алгоритмов оптимизации, который способствует исследованию различных решений и улучшению качества найденных оптимальных решений.

Обновление популяции в метаэвристических алгоритмах оптимизации представляет собой процесс изменения состава и свойств особей или решений в текущей популяции. Этот шаг обновления популяции играет важную роль в процессе поиска оптимального решения, позволяя алгоритму исследовать пространство поиска более эффективно. В различных метаэвристических алгоритмах обновление популяции может происходить по-разному. Например, в генетических алгоритмах новая популяция может быть создана путем комбинирования лучших особей текущей популяции (селекция), применения к ним операторов скрещивания и мутации. В других алгоритмах, таких как муравьиные алгоритмы или ройовые алгоритмы, обновление популяции может включать в себя перемещение решений или особей в пространстве поиска на основе специальных правил поведения. Цель обновления популяции заключается в том, чтобы сохранять разнообразие в популяции, избегать застревания в локальных оптимумах и продолжать исследование пространства поиска для нахождения наилучшего решения. Правильное обновление популяции способствует улучшению качества найденного оптимального решения и повышению эффективности метаэвристического алгоритма оптимизации.

2.1.1 Обзор метаэвристик

Bio Based:

Biogeography-Based Optimization (BBO) [16] — это метаэвристический алгоритм оптимизации, который моделирует биогеографические процессы, такие как миграция, мутация и колонизация,

для решения задач оптимизации. BBO вдохновлен теорией биогеографии, которая изучает распределение организмов в пространстве и времени.

Инициализация: В начале работы BBO исходная популяция решений (островов) инициализируется случайным образом в пространстве поиска решений. Каждый остров представляет собой набор параметров, которые определяют потенциальное решение задачи оптимизации.

Миграция: Основной идеей BBO является процесс миграции, при котором решения (организмы) перемещаются между островами в соответствии с их приспособленностью. Более приспособленные решения имеют больше шансов мигрировать на другие острова.

Мутация и колонизация: В BBO также присутствуют процессы мутации и колонизации, которые позволяют вносить разнообразие в популяцию решений и обеспечивать исследование новых областей пространства поиска.

Обновление популяции: после каждого цикла миграции, мутации и колонизации происходит обновление популяции на каждом острове на основе принципа отбора по приспособленности. Это позволяет сохранять лучшие решения и улучшать качество популяции с каждой итерацией.

Критерий останова: BBO продолжает свою работу до достижения критерия останова, такого как достижение заданной точности или превышение максимального числа итераций.

Применение: Biogeography-Based Optimization может быть успешно применен для решения различных задач оптимизации в областях, таких как инженерия, экономика, биология, информационные технологии и другие.

Преимущества и недостатки: Преимуществами BBO являются способность к глобальной оптимизации, устойчивость к локальным оптимумам, возможность работы с дискретными и непрерывными пространствами поиска. Однако недостатками могут быть необходимость настройки параметров алгоритма и высокая вычислительная сложность.

Invasive Weed Optimization (IWO) [17] — это метаэвристический алгоритм оптимизации, который моделирует процессы распространения сорняков в природной среде для решения задач оптимизации. IWO вдохновлен поведением инвазивных растений, которые способны быстро распространяться и захватывать новые территории.

Инициализация: В начале работы IWO исходная популяция решений (семян сорняков) инициализируется случайным образом в пространстве поиска решений. Каждое семя представляет собой потенциальное решение задачи оптимизации.

Распространение: Основной идеей IWO является процесс распространения, при котором семена (решения) распространяются по окружающей среде, осуществляя поиск более благоприятных мест для роста и размножения. Этот процесс позволяет исследовать различные области пространства поиска.

Конкуренция и адаптация: В процессе распространения семена сорняков конкурируют за доступ к ресурсам и могут адаптироваться к изменяющимся условиям среды. Аналогично, в IWO решения конкурируют за приспособленность и могут изменяться в процессе оптимизации.

Обновление популяции: после каждого цикла распространения происходит обновление популяции на основе принципа отбора по приспособленности. Лучшие решения сохраняются, а менее приспособленные могут быть заменены новыми семенами.

Критерий останова: IWO продолжает свою работу до достижения критерия останова, такого как достижение заданной точности или превышение максимального числа итераций.

Применение: Invasive Weed Optimization может быть успешно применен для решения различных задач оптимизации, таких как функциональная оптимизация, задачи машинного обучения, планирование и др.

Преимущества и недостатки: Преимуществами IWO являются способность к глобальной оптимизации, быстрая сходимость к оптимальному решению, простота реализации и возможность работы с различными типами задач. Однако недостатками могут быть необходимость настройки параметров алгоритма и возможные проблемы с локальными оптимумами.

Evolutionary Based:

Differential Evolution (DE) [18] является эволюционным оптимизационным методом, который используется для решения задач оптимизации. DE был предложен в 1995 году Стормом и Прайсом и с тех пор стал одним из наиболее популярных методов оптимизации.

DE основан на эволюционном подходе, который имитирует процесс естественного отбора в природе. Основная идея DE заключается в том, что он создает и изменяет популяцию кандидатов-решений, называемых "индивидами", чтобы находить оптимальное решение задачи оптимизации.

DE работает следующим образом:

1. Инициализация популяции: начальная популяция индивидов генерируется случайным образом.

2. Мутация: для каждого индивида в популяции производится мутация, которая изменяет его значение с помощью различных стратегий мутации.

3. Скрещивание: новые индивиды формируются путем скрещивания между текущими индивидами и их мутантами.

4. Отбор: новая популяция формируется путем отбора лучших индивидов из текущей популяции и новообразованных индивидов.

5. Повторение: шаги мутации, скрещивания и отбора повторяются до достижения критерия останова (например, достижения определенного количества итераций или достижения заданного уровня сходимости).

DE имеет несколько параметров, таких как размер популяции, вероятности мутации и скрещивания, а также стратегии мутации. Выбор этих параметров может существенно влиять на производительность алгоритма.

DE широко применяется для решения различных задач оптимизации, таких как поиск глобального экстремума функций, обучение нейронных сетей, оптимизация параметров моделей машинного обучения и другие. Он обладает хорошей способностью к сходимости и высокой эффективностью при решении сложных задач оптимизации.

Human Based:

Culture Algorithm (CA) [19] — это метаэвристический алгоритм оптимизации, который моделирует социокультурные процессы для решения задач оптимизации. CA вдохновлен социальными и культурными аспектами человеческого общества, такими как обмен знаниями, обучение, эволюция и коллективное поведение.

Популяция: В CA популяция состоит из индивидов, которые представляют собой потенциальные решения задачи оптимизации. Каждый индивид хранит свое генетическое кодирование (решение) и информацию о своем культурном опыте.

Культурный опыт: Одной из ключевых концепций CA является культурный опыт, который представляет собой накопленное знание и опыт популяции. Индивиды могут обмениваться информацией, учиться друг у друга и принимать решения на основе культурного контекста.

Обучение и эволюция: В CA индивиды могут обучаться на основе своего собственного опыта и опыта других членов популяции. Это позволяет эффективно использовать накопленные знания для улучшения качества решений и приспособления к изменяющимся условиям.

Культурные алгоритмы: В рамках CA используются различные культурные алгоритмы, которые определяют способы обмена информацией, обучения и эволюции в популяции. Примерами таких алгоритмов могут быть алгоритмы обучения, имитации и эволюции.

Применение: Culture Algorithm может быть успешно применен для решения различных задач оптимизации, таких как функциональная оптимизация, задачи машинного обучения, планирование и т. д. Благодаря

учету социокультурных аспектов, CA может обладать высокой эффективностью и способностью к адаптации к сложным условиям.

Преимущества и недостатки: Преимуществами Culture Algorithm являются способность к коллективному интеллекту, учет культурного контекста при принятии решений, возможность обмена знаниями и опытом между индивидами. Однако недостатком может быть сложность реализации алгоритма и необходимость настройки параметров под конкретную задачу.

Forensic-Based Investigation Optimization (FBIO) [20] — этот алгоритм основан на принципах и методах, используемых в судебной экспертизе, криминалистике и других областях судебной деятельности.

Принцип работы FBIO основан на имитации процессов и методов, которые применяются при проведении судебных исследований. Алгоритм использует механизмы поиска, оценки и выбора оптимальных решений на основе знаний и опыта, накопленных в области судебной экспертизы.

FBIO может быть применен в различных сферах правоохранительной деятельности, включая уголовное правосудие, борьбу с коррупцией, финансовые преступления, кибербезопасность и другие области, где требуется выявление и раскрытие преступлений.

Math Based:

Gradient-Based Optimizer (GBO) [21] — GBO, основанный на градиентном методе Ньютона, использует два основных оператора: правило градиентного поиска (GSR) и локальный экранирующий оператор (LEO), а также набор векторов для исследования пространства поиска. В GSR используется метод, основанный на градиенте, для усиления тенденции к поиску и ускорения скорости сходимости для достижения лучших позиций в пространстве поиска. LEO позволяет предлагаемому алгоритму GO выйти за пределы локальных оптимумов..

Градиент функции цели: Градиент функции цели — это вектор, который указывает направление наискорейшего возрастания функции. В

случае минимизации функции цели, градиент будет указывать направление наискорейшего убывания функции.

Итеративный процесс: Gradient-Based Optimizer работает пошагово, изменяя параметры решения на каждом шаге в направлении, противоположном градиенту функции. Этот процесс повторяется до тех пор, пока не будет достигнута заданная точность или установлен критерий останова.

Методы оптимизации: существует несколько методов оптимизации, которые могут быть использованы в Gradient-Based Optimizer, такие как градиентный спуск, стохастический градиентный спуск, метод Ньютона и другие. Каждый из этих методов имеет свои особенности и применимость в различных задачах оптимизации.

Применение: Gradient-Based Optimizer широко применяется в машинном обучении, искусственном интеллекте, оптимизации функций, обратном распространении ошибки в нейронных сетях и других областях, где требуется решение задач оптимизации.

Преимущества и недостатки: Преимуществами Gradient-Based Optimizer являются эффективность и скорость сходимости к оптимальному решению. Однако недостатками могут быть проблемы с локальными оптимумами, неустойчивость к шуму и вычислительная сложность в случае больших размерностей.

Swarm Based:

Artificial Bee Colony (ABC) [22] — это метаэвристический алгоритм оптимизации, который моделирует поведение пчелиной колонии для решения задач оптимизации. ABC был впервые предложен в 2005 году и быстро стал популярным методом оптимизации благодаря своей простоте и эффективности.

Пчелиная колония: ABC моделирует поведение пчелиной колонии, состоящей из трех типов пчел: рабочих пчел, смотрителей улья и источников

пищи. Рабочие пчелы и источники пищи отвечают за поиск оптимальных решений, а смотрители улья контролируют качество найденных решений.

Решение: В ABC каждое решение представляется в виде позиции в пространстве поиска, которое соответствует потенциальному оптимальному значению переменных задачи оптимизации. Рабочие пчелы и источники пищи исследуют это пространство для нахождения лучших решений.

Этапы работы: ABC состоит из нескольких этапов, включая инициализацию популяции пчел, фазу поиска, фазу обновления и фазу оценки качества решений. На каждом этапе пчелы взаимодействуют друг с другом и средой для нахождения оптимального решения.

Поиск пищи: Рабочие пчелы и источники пищи исследуют пространство поиска, обмениваются информацией о найденных решениях и выбирают лучшие решения для дальнейшего улучшения. Этот процесс напоминает поведение настоящей пчелиной колонии при поиске цветущих цветов.

Обновление: В процессе работы ABC пчелы обновляют свои решения на основе информации, полученной от других пчел и результатов оценки качества найденных решений. Это позволяет улучшать качество решений и приближаться к оптимальному значению задачи оптимизации.

Применение: Artificial Bee Colony может быть успешно применен для решения различных задач оптимизации, таких как функциональная оптимизация, задачи машинного обучения, задачи коммивояжера и другие. ABC обладает хорошей способностью к эксплорации и эксплойтации пространства поиска решений.

Преимущества и недостатки: Преимуществами ABC являются простота реализации, эффективность в поиске оптимальных решений, способность к адаптации к различным задачам. Однако недостатком может быть необходимость настройки параметров алгоритма под конкретную задачу и возможность застревания в локальных оптимумах.

Ant Colony Optimization (ACO) [23] — это метаэвристический алгоритм оптимизации, вдохновленный поведением муравьев при поиске пищи. ACO был предложен в начале 1990-х годов и с тех пор активно развивается и применяется для решения различных задач оптимизации, таких как коммивояжерская проблема, задача раскроя и другие.

Инспирация от природы: ACO моделирует поведение муравьев, которые обладают способностью находить кратчайший путь к источнику пищи благодаря феромонам, оставляемым на своем пути. Муравьи обмениваются информацией через феромоны, что позволяет им эффективно координировать свои действия и находить оптимальные решения.

Решение: В ACO каждое решение представляется в виде маршрута или пути в пространстве поиска, который соответствует оптимальному значению переменных задачи оптимизации. Муравьи исследуют пространство поиска, строят решения на основе феромонов и выбирают оптимальные пути.

Основные компоненты: ACO состоит из нескольких ключевых компонентов, таких как феромоны, эвристика, правила обновления феромонов и механизм выбора путей. Феромоны используются для передачи информации о качестве решений, а эвристика помогает муравьям принимать решения на основе локальной информации.

Поиск пути: В процессе работы ACO муравьи исследуют пространство поиска, перемещаясь по графу решений и обновляя феромоны на пройденных путях. Муравьи предпочитают выбирать пути с более высокой концентрацией феромонов, что способствует нахождению оптимального решения.

Обновление феромонов: Важным аспектом ACO является обновление феромонов после каждого цикла поиска. Феромоны испаряются с течением времени, что позволяет алгоритму избегать застревания в локальных оптимумах и искать новые пути к глобальному оптимуму.

Применение: Ant Colony Optimization может быть успешно применен для решения задач комбинаторной оптимизации, таких как задача

коммивояжера, задача раскроя, задача размещения и другие. ACO обладает способностью к адаптации к различным типам задач и эффективно находит оптимальные решения в сложных пространствах поиска.

Преимущества и недостатки: Преимуществами ACO являются способность к нахождению глобального оптимума, высокая параллелизуемость, а также простота реализации и интерпретации результатов. Однако недостатками могут быть необходимость в большом количестве вычислительных ресурсов и сложность настройки параметров алгоритма.

Harris Hawks Optimization (HHO) [24] — это метаэвристический алгоритм оптимизации, вдохновленный поведением хищных птиц, таких как ястребы Харриса. HHO был предложен в 2019 году и быстро привлек внимание исследователей благодаря своей эффективности и способности к решению сложных задач оптимизации.

Инспирация от природы: HHO моделирует поведение харрисовых ястребов при охоте на добычу. В процессе охоты ястребы сотрудничают, обмениваются информацией и координируют свои действия для успешного поимки добычи. Эти принципы вдохновили создателей HHO на разработку алгоритма оптимизации.

Решение: В HHO каждое решение представляется в виде позиции в пространстве поиска, которое соответствует потенциальному оптимальному значению переменных задачи оптимизации. Популяция решений состоит из "ястребов", которые исследуют пространство поиска для нахождения лучших решений.

Основные компоненты: HHO состоит из нескольких ключевых компонентов, таких как фаза поиска добычи, фаза обновления, фаза совместного поиска и фаза оценки качества решений. В каждой фазе ястребы взаимодействуют друг с другом и средой для достижения оптимального решения.

Поиск добычи: В фазе поиска добычи ястребы исследуют пространство поиска, перемещаясь к лучшим решениям и обмениваясь информацией о найденных позициях. Этот процесс напоминает стратегии охоты харрисовых ястребов, которые координированно действуют для достижения цели.

Обновление: В процессе работы HHO ястребы обновляют свои позиции на основе информации, полученной от других "ястребов" и результатов оценки качества найденных решений. Это позволяет улучшать качество решений и приближаться к оптимальному значению задачи оптимизации.

Применение: Harris Hawks Optimization может быть успешно применен для решения различных задач оптимизации, таких как функциональная оптимизация, задачи машинного обучения, задачи планирования и другие. HHO обладает способностью к быстрой сходимости к оптимальному решению и адаптации к различным типам задач.

Преимущества и недостатки: Преимуществами HHO являются высокая эффективность в поиске оптимальных решений, способность к совместной работе популяции "ястребов", а также возможность быстрой адаптации к изменяющимся условиям задачи. Однако недостатком может быть необходимость настройки параметров алгоритма под конкретную задачу.

Particle Swarm Optimization (PSO) [25] — это метаэвристический алгоритм оптимизации, инспирированный поведением стаящихся птиц или роящихся рыб. В PSO решение задачи оптимизации представлено как "частица", которая движется в пространстве поиска решений с целью нахождения оптимального значения целевой функции. Каждая частица имеет свое положение и скорость, которые изменяются в зависимости от ее собственного опыта и опыта других частиц в стае.

Инициализация: В начале работы алгоритма каждая частица инициализируется случайным образом в пространстве поиска решений. Каждая частица также имеет свое собственное лучшее известное положение (локальный оптимум) и лучшее известное положение стаи (глобальный оптимум).

Движение частиц: На каждом шаге PSO каждая частица обновляет свое положение и скорость в соответствии с формулами, учитывающими ее текущее положение, скорость, лучший локальный оптимум и лучший глобальный оптимум в стае.

Обновление лучших значений: После обновления положения каждая частица сравнивает свое новое значение целевой функции с лучшим известным значением (локальным и глобальным) и обновляет их при необходимости.

Критерий останова: PSO продолжает свою работу до тех пор, пока не будет достигнут критерий останова, такой как достижение заданной точности или превышение максимального числа итераций.

Применение: Particle Swarm Optimization широко применяется в различных областях, таких как машинное обучение, инженерия, экономика, биология и другие, для решения задач оптимизации, таких как поиск глобального оптимума функции, кластеризация данных, обучение нейронных сетей и др.

Преимущества и недостатки: Преимуществами PSO являются простота реализации, способность находить глобальный оптимум в многомерных пространствах и высокая скорость сходимости. Однако недостатками могут быть проблемы с локальными оптимумами, зависимость от параметров алгоритма и неэффективность в случае сложных задач.

2.1.2 Тестирование метаэвристик

Для тестирования метаэвристик были выбраны специальные функции, предназначенные под их тестирование: Функция Изама, Функция Экли, Функция Экли №3, Функция Кина, Функция Михалевича, Функция Розенброка, Негладкая многовершинная функция, Функция Брауна, Функция Леви №13, Функция Птица, Функция Швефеля, Функция Брента, Функция Деккерса-Аартса, Функция Син-Ше Янга №4, Функция Яйца в держателе, Функция Шуберта, Функция Шаффера, Функция Шуберта №4, Функция Волнения, Функция Кросс-ин-трей. Эти функции были специально

разработаны для тестирования алгоритмов оптимизации с целью поиска глобального оптимума. Также были написаны композитные функции, состоящие из нескольких функций, где каждая из которых имеет какой-то вес (коэффициент). Рассмотрим часть из этих функций подробно.

Функция Аклея №3 — это математическая функция, широко используемая в оптимизации. Она также определена на двумерном пространстве и имеет вид

$$f(x,y) = -200e^{-0.2\sqrt{x^2+y^2}} - 5e^{\cos 3x+\sin 3y}$$

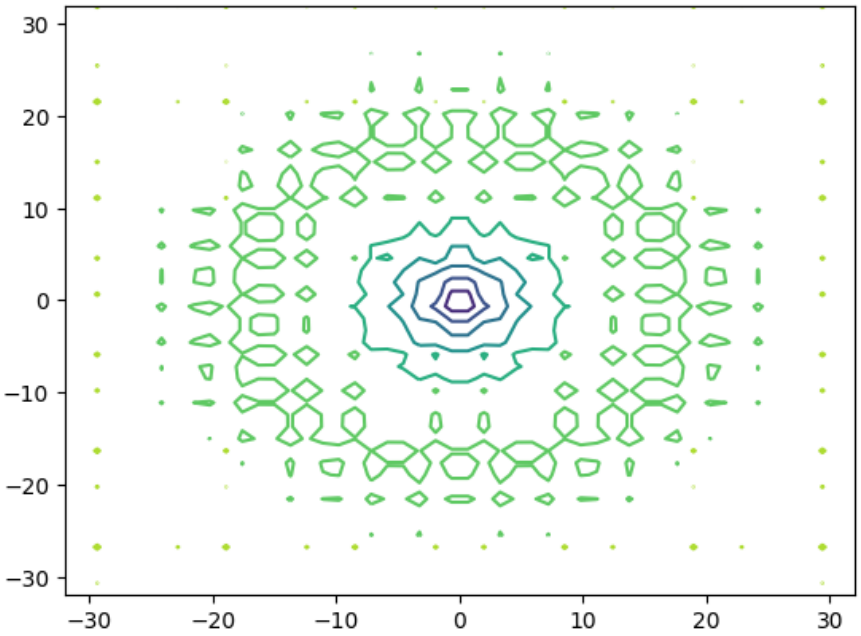


Рисунок 2.1 Функция Аклея №3 (Вид сверху).

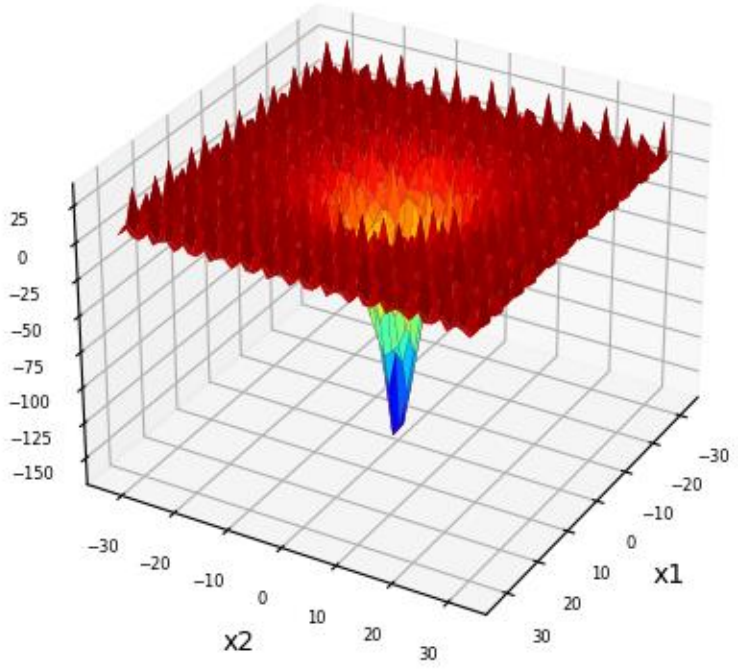


Рисунок 2.2 Функция Аклея №3.

Функция Леви №13 (Levi №13 Function) — это функция многих переменных, которая также используется в задачах оптимизации для проверки эффективности алгоритмов оптимизации. Функция Леви №13 имеет один глобальный оптимум и несколько локальных оптимумов, что делает ее сложной для оптимизации.

$$f(x,y) = sin^2(3\pi x) + (x-1)^2\left(1 + sin^2(3\pi y)\right) + (y-1)^2(1 + sin^2(2\pi y))$$

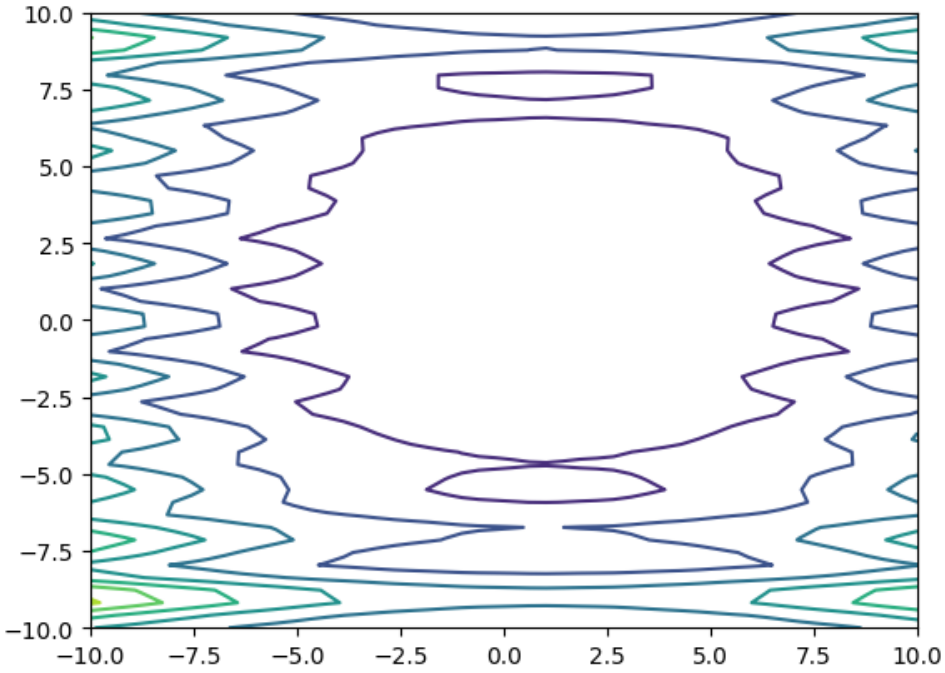


Рисунок 2.3 Функция Леви N.13 (Вид сверху).

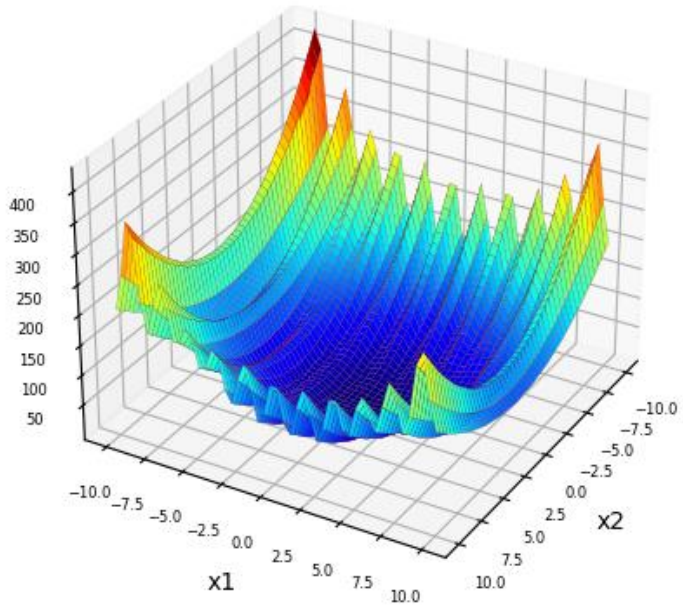


Рисунок 2.4 Функция Леви N.13.

Функция Шаффера (Schaffer function) — это негладкая многовершинная функция, которая часто используется в задачах оптимизации и тестирования алгоритмов оптимизации. Функция Шаффера имеет несколько глобальных минимумов и локальных минимумов, что

делает ее хорошим тестовым примером для оценки производительности оптимизационных методов.

Функция Шаффера обычно представляется в виде:

$$f(x,y) = 0.5 + \frac{sin^2(x^2 - y^2) - 0.5}{(1 + 0.001(x^2 + y^2))^2}$$

где x и y — переменные, которые определяют положение точки в пространстве, а f(x, y) — значение функции в этой точке.

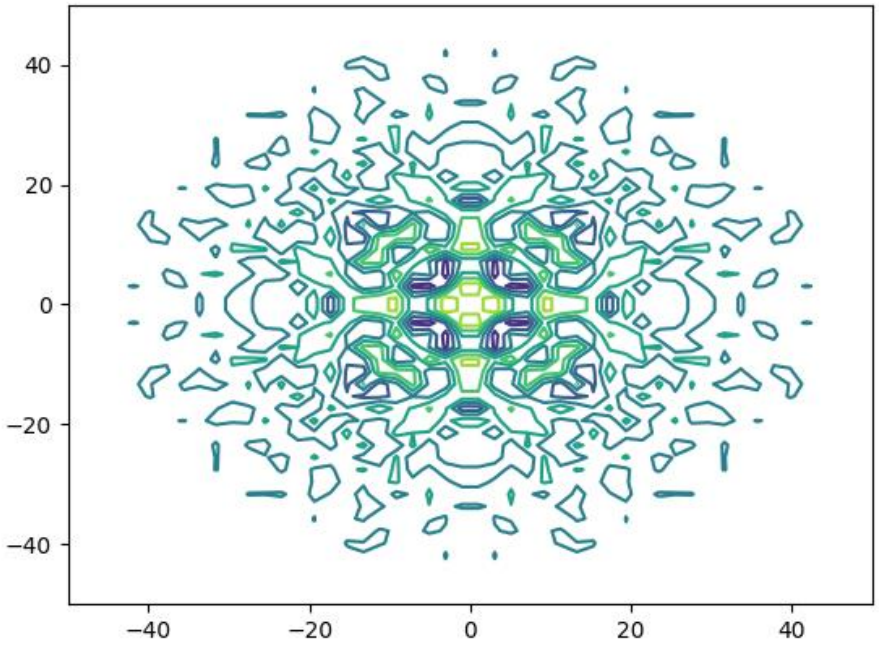


Рисунок 2.5 Функция Шаффера (Вид сверху).

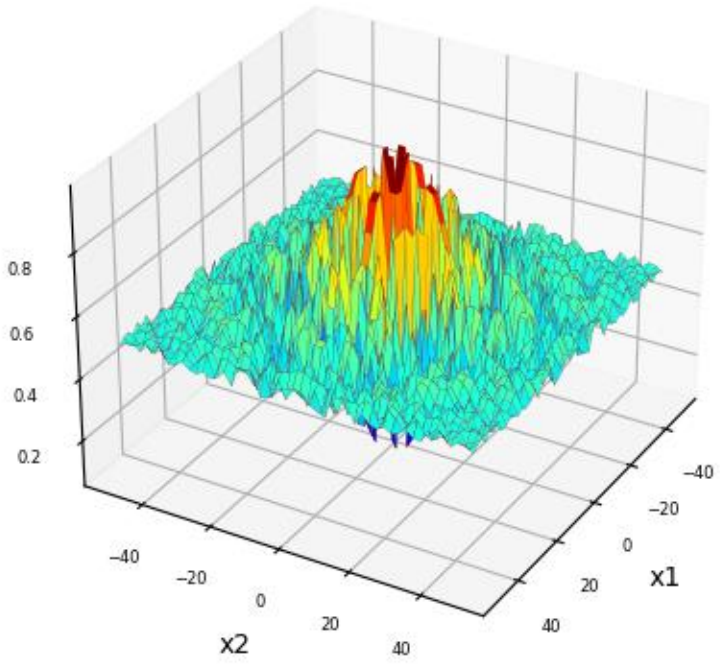


Рисунок 2.6 Функция Шаффера.

Из-за своей сложной формы и наличия нескольких минимумов функция Шаффера часто используется для проверки эффективности различных методов оптимизации, таких как генетические алгоритмы, методы градиентного спуска и метаэвристические алгоритмы.

Композитная функция Шаффера и Волнения. Данная функция была создана путем объединения функции Шафера и функции Волнения с коэффициентами 0.2 и 0.8 соответсвенно.

$$f(x,y) = 0.2\left(0.5 + \frac{sin^2(x^2 - y^2) - 0.5}{\left(1 + 0.001(x^2 + y^2)\right)^2}\right) + 0.8\left(\frac{1 + \cos(12\sqrt{x^2 + y^2})}{0.5(x^2 + y^2) + 2}\right)$$

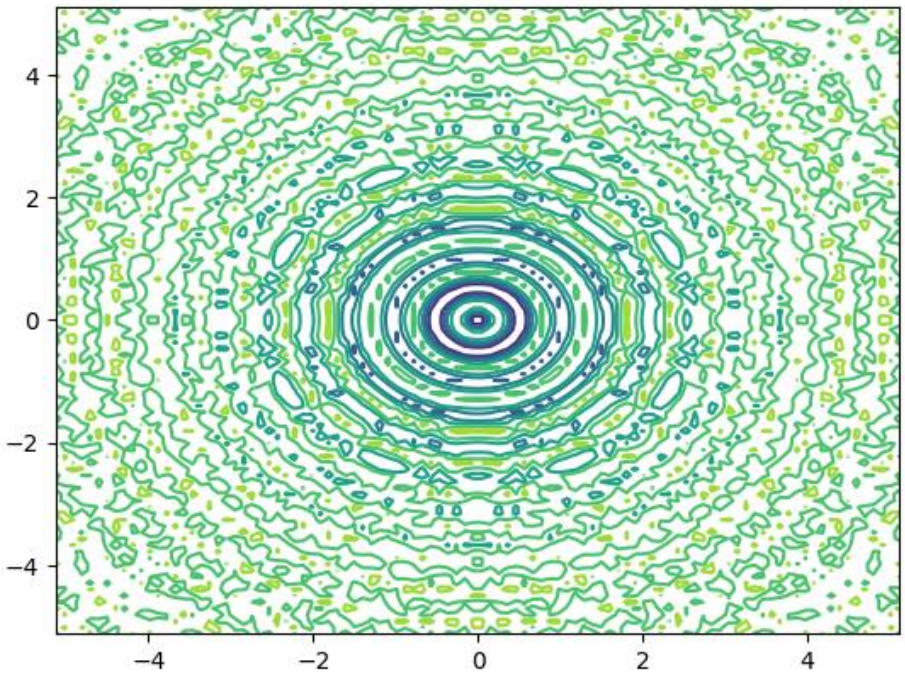


Рисунок 2.7 Композитная функция Шаффера и Волнения (Вид сверху).

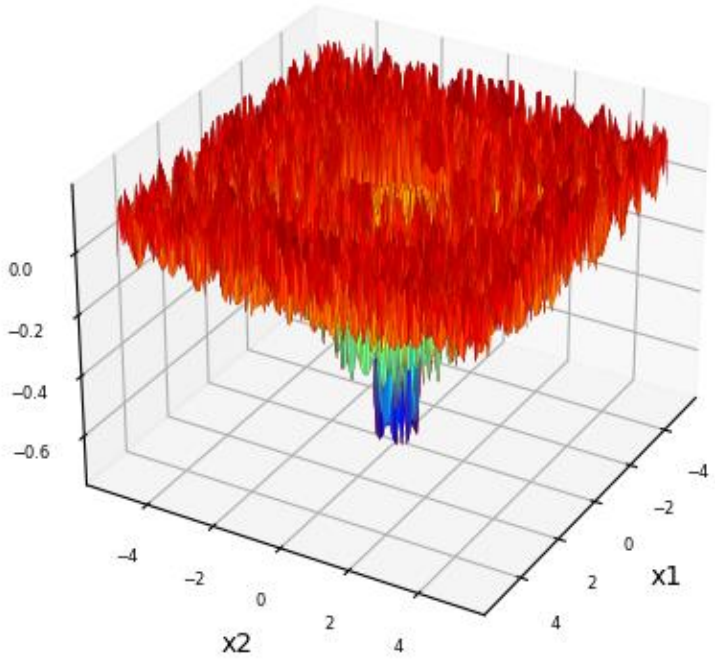


Рисунок 2.8 Композитная функция Шаффера и Волнения.

Проведя тесты над реализованными в рамках работы метаэвристиками на тесовых функциях и сделав 77 тысяч измерений можно увидеть следующие результаты:

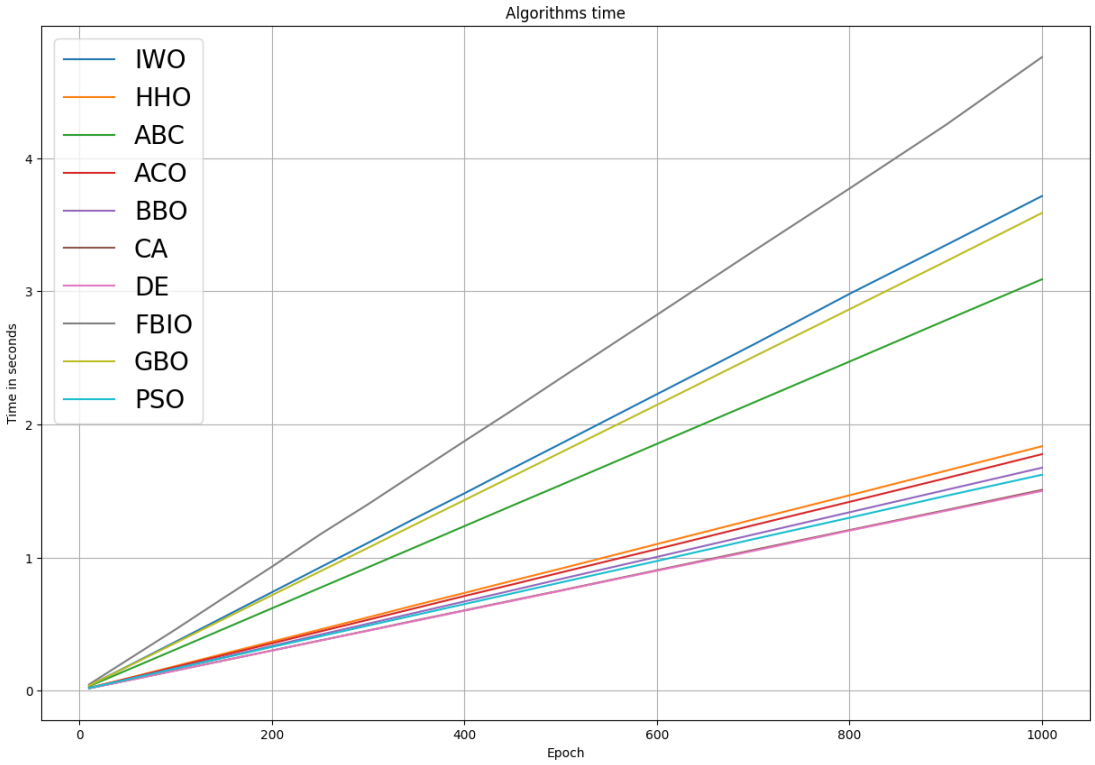


Рисунок 2.9 График зависимости времени работы метаэвристик от пройденных эпох популяции.

На данном графике можно увидеть зависимость времени работы метаэвристик от роста числа эпох в их популяциях. Наилучшее время показывают эвристики PSO, DE и CA, а наихудшие результаты по скорости работы у FBIO, IWO и GBO.

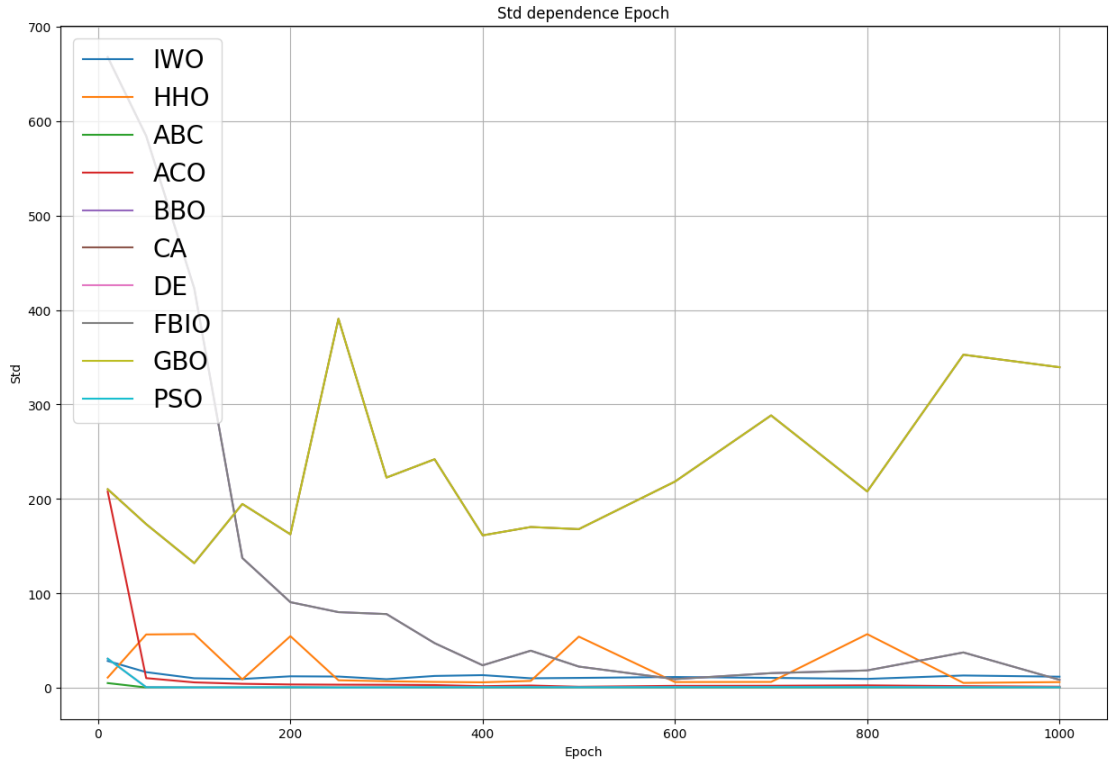


Рисунок 2.10 График зависимости стандартного отклонения от числа эпох популяции.

Данный график показывает зависимость стандартного отклонения от глобального оптимума от числа эпох в популяциях метаэвристиках. На графике хорошо видно, что самые точные результаты дали эвристики ABC, PSO и IWO, хуже всех с тестовыми функциями справились GBO и FIBO.

Таблица 1. Результаты работы тестирования метаэвристик

Name_evr	Time	Best_Fit	Std	Optimum
ABC	0.6179	-1044.358	0.5874	-1044.5096
ACO	0.3548	-1041.099	3.7927	-1044.5096
BBO	0.3349	-953.6668	90.8623	-1044.5096
CA	0.3010	-881.8642	162.6671	-1044.5096
DE	0.3002	-1044.0399	0.9056	-1044.5096
FBIO	0.9305	-1044.3468	0.5987	-1044.5096
GBO	0.7165	-1044.1065	0.8389	-1044.5096
HHO	0.3677	-990.0099	54.9138	-1044.5096
IWO	0.7389	-1032.5178	12.4277	-1044.5096
PSO	0.3278	-1030.7730	13.7869	-1044.5096

Данная таблица показывает среднее время, стандартное отклонение от глобального оптимума, также сам оптимум и лучшее решение, найденное метаэвристиками.

Пусть с композитными функциями плохо справились метаэвристики BBO, CA, HHO, но на обычных функциях они показали свою отличную работоспособность и обоснованность в их использовании в исследовании. Однако можно выделить наилучшие по соотношению скорости и точности решения задач метаэвристики ABC и DE.

Код реализации, а также расчеты находятся в публичном репозитории на github [27]

2.2 Встраивание информации в цифровые изображения

В данной работе используются различные ансамбли метаэвристик для проверки их эффективности на стеганографическом алгоритме [26]. Данный алгоритм имеет достаточно высокие показатели по соотношению сигнал шум, что показывает его способность скрывать информацию так, чтобы она была полностью незаметна для человеческого глаза. Также данный алгоритм является алгоритмом безошибочного извлечения информации, а для самого декодирования не требуется ни закрытого, ни открытого ключа. Эти особенности стеганографического алгоритма показывают, что в случае, например нарушения безопасности Men-In-Middle посмотрев на изображение человек не поймет, что в ней спрятан водный знак и сокрытая информация. Также из-за особенности встраивания информации в данном алгоритме он является криптостойким для анализа на сокрытую информацию.

2.2.1 Общие сведения

Встраивание информации в цифровые изображения, также известное как стеганография, является техникой сокрытия данных в цифровых изображениях таким образом, чтобы они были незаметны для человеческого глаза. Эта техника используется для различных целей, включая защиту авторских прав, аутентификацию и секретную связь. Существует множество различных алгоритмов встраивания информации в цифровые изображения,

каждый из которых имеет свои преимущества и недостатки. Вот обзор некоторых наиболее популярных алгоритмов:

Алгоритм наименьшего значащего бита (LSB) является одним из самых простых и распространенных алгоритмов встраивания информации. Он работает путем замены наименее значимого бита каждого пикселя в изображении битом данных. Этот метод прост в реализации и обеспечивает относительно высокую емкость встраивания, но он также уязвим для обнаружения и удаления.

Алгоритм прямого разнесения спектра (DSSS) распространяет данные по всему изображению, а не сосредотачивает их в определенных областях. Это делает его более устойчивым к обнаружению и удалению, чем алгоритм LSB. Однако алгоритм DSSS также имеет более низкую емкость встраивания, чем алгоритм LSB.

Алгоритмы сжатия с потерями, такие как JPEG и MPEG, могут использоваться для встраивания информации в цифровые изображения путем внесения небольших изменений в сжатые данные. Эти изменения обычно незаметны для человеческого глаза, но их можно обнаружить с помощью специальных инструментов. Алгоритмы сжатия с потерями обеспечивают относительно высокую емкость встраивания, но они также могут привести к некоторому снижению качества изображения.

Алгоритм квантования работает путем изменения значений пикселей в изображении таким образом, чтобы они соответствовали определенным уровням квантования. Эти уровни квантования затем используются для встраивания данных. Алгоритм квантования обеспечивает хорошую емкость встраивания и устойчивость к обнаружению, но он также может привести к некоторому снижению качества изображения.

Алгоритм фазовой модуляции работает путем изменения фазы пикселей в изображении таким образом, чтобы они соответствовали данным. Этот метод обеспечивает высокую емкость встраивания и устойчивость к

обнаружению, но он также может привести к некоторому снижению качества изображения.

Выбор алгоритма встраивания информации в цифровые изображения зависит от конкретных требований приложения. Для приложений, требующих высокой емкости встраивания, могут подойти алгоритмы LSB или сжатия с потерями. Для приложений, требующих высокой устойчивости к обнаружению, могут подойти алгоритмы DSSS, квантования или фазовой модуляции. Помимо этих популярных алгоритмов, существует множество других алгоритмов встраивания информации в цифровые изображения, каждый из которых имеет свои собственные уникальные характеристики. Выбор наилучшего алгоритма для конкретного приложения требует тщательного рассмотрения требований к емкости встраивания, устойчивости к обнаружению и качеству изображения.

2.2.2 Алгоритм встраивания информации в фазовый спектр дискретного преобразования Фурье

Алгоритм [26] делит изображение на блоки размером 8 × 8 пикселей, которые не пересекаются. Далее для каждого блока производится дискретное преобразование Фурье (DFT). Для встраивания используется фазовая составляющая блока DFT. Важно отметить, что фазовый спектр DFT является нечетной функцией, поэтому при изменении элемента фазового спектра φ(u,v) необходимо симметрично изменить соответствующий элемент.

$$\varphi(-u,-v) = -\varphi(\mathrm{u},\mathrm{v}), \mathrm{u} = \overline{0,7}, v = \overline{0,7}$$

Область встраивания включает 21 блок размером 8 × 8 пикселей, охватывая все высокочастотные и среднечастотные компоненты спектра DFT. Размер этой области был определен экспериментально для оптимальной емкости встраивания. Использование наиболее однородных блоков изображения с амплитудой ниже порогового значения Acrit для встраивания информации в фазовые значения коэффициентов DFT. Фазовые значения

находятся в диапазоне $(-\pi; \pi]$, и для встраивания каждого бита сообщения используются два значения $\varphi 0$ и $\varphi 1$, где $\varphi 0 = -\varphi 1$.

Характерной особенностью данного алгоритма является использование итеративного процесса встраивания информации. Этот подход направлен на коррекцию ошибок извлечения, возникших из—за округления данных. В процессе итеративного встраивания информации происходит извлечение данных, проверка ошибок извлечения после встраивания фрагмента секретного сообщения в блок коэффициентов дискретного преобразования Фурье (DFT). При извлечении данных выполняется обратное DFT, округление значений до целых пикселей и повторное DFT. Эти процедуры могут привести к искажению фазовых значений коэффициентов. Поэтому при извлечении используется интервал значений, ширина которого определяется параметром алгоритма ε. Формула для извлечения имеет следующий вид:

$$b = \begin{cases} 0, \text{если } \varphi'' \in (\varphi_0 - \varepsilon, \varphi_0 + \varepsilon) \\ 1, \text{если } \varphi'' \in (\varphi_1 - \varepsilon, \varphi_1 + \varepsilon) \end{cases}$$

В алгоритме используется итеративный процесс для исправления ошибок извлечения информации. Если ошибки не удается исправить в течение заданного числа итераций, блок считается пустым. Для минимизации количества переменных фазового спектра применяется оптимизация. В некоторых случаях требуется много итераций для безошибочного извлечения информации, что может уменьшить незаметность внедрения. Выбор длины фрагмента сообщения и его расположение в области встраивания влияет на эффективность алгоритма. Основная идея заключается в преобразовании последовательности фазовых значений для оптимального встраивания информации.

Описанный алгоритм позволяет учитывать начальные значения элементов фазового спектра DFT. В результате уменьшается количество искажений в конечном стего-изображении.

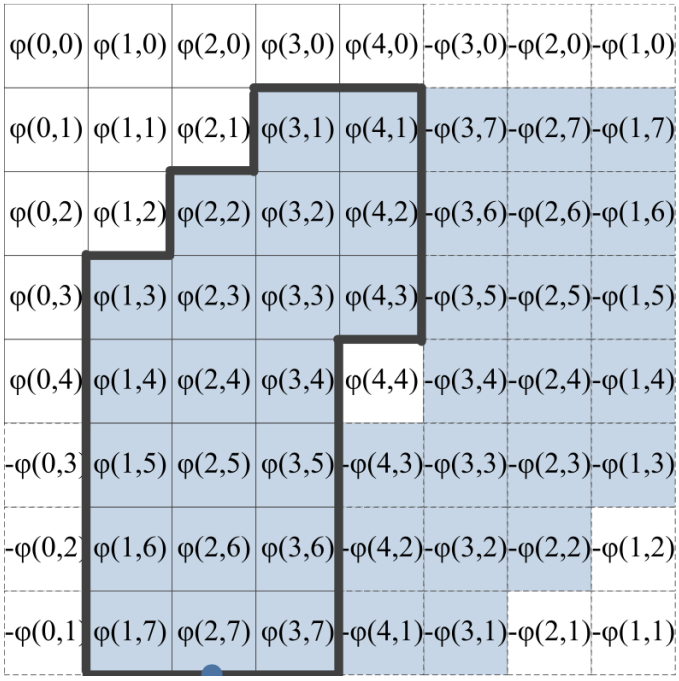


Рисунок 2.11 Область встраивания алгоритма

Результаты экспериментов демонстрируют высокий уровень незаметности и хорошее значение емкости. Соответственно мтеаэвристики используются для поиска наилучших значений в выделенной жирной линией области и дальнейшей работы с ними. Целевая функция для оптимизации:

$$F = k_C C_f + k_{PSNR} PSNR_f + k_E E_f$$

где $k_C -$ весовой коэффициент емкости ечейки, $k_{PSNR} -$ весовой коэффициент PSNR, $k_E -$ весовой коэффициент ошибок извлечения.

$C_f -$ вместимость ячейки, находится по формуле:

$$C_f = \frac{C_{block}}{C_{max}}$$

$C_{block} -$ вместимость анализируемого блока, $C_{max} - 21$ как видно из рисунка 2.11.

$E_f -$ Число ошибок извлечения информации из области встраивания.

$PSNR_f -$ отношение сигнал/шум в области встраивания, расчитывается по формуле:

$$PSNR_f = 10 * \log_{10}(\frac{255^2}{MSE})$$

где $MSE = \frac{1}{8*8}\sum_{i=1}^{8*8}(I_i - S_i)^2$

I_i – значение пикселей оригинального изображения, S_i – значение пикселей стегоизображения.

3 Разработка подхода к построению ансамбля метаэвристик

Ансамбль метаэвристик — это концепция в области искусственного интеллекта и оптимизации, которая объединяет несколько различных метаэвристических методов для решения сложных задач оптимизации. В случае ансамбля метаэвристик, несколько различных методов метаэвристической оптимизации (например, генетические алгоритмы, муравьиные алгоритмы, симуляция отжига и др.) используются одновременно или последовательно для решения задачи оптимизации. Каждый из этих методов может обладать своими сильными и слабыми сторонами, и комбинирование их в ансамбль позволяет улучшить качество найденного решения. Применение ансамбля метаэвристик позволяет увеличить вероятность нахождения оптимального решения за счет комбинирования различных методов и подходов к оптимизации. Такой подход широко применяется в различных областях, таких как машинное обучение, инженерия, финансы и другие, где требуется решение сложных задач оптимизации.

Ансамбли метаэвристик обычно состоят из нескольких базовых метаэвристических методов, таких как генетические алгоритмы, муравьиные алгоритмы, методы оптимизации роя частиц и др. Каждый из этих методов имеет свои сильные и слабые стороны, и комбинирование их в ансамбль позволяет получить более устойчивое и эффективное решение задачи оптимизации. Принцип работы ансамблей метаэвристик заключается в том, что каждый метод вносит свой вклад в процесс оптимизации, а ансамбль в целом способен обнаруживать разнообразные решения и избегать застревания в локальных оптимумах. Обычно ансамбли метаэвристик используются для решения сложных задач оптимизации в различных областях, таких как инженерия, экономика, биоинформатика и другие. Преимущества ансамблей метаэвристик включают повышенную надежность и устойчивость к различным типам задач, способность к исследованию большего пространства решений и улучшение качества найденных решений.

Однако создание и настройка ансамблей метаэвристик может быть нетривиальной задачей, требующей глубокого понимания каждого метода и их взаимодействия.

В данной работе используются три идеи для реализации ансамблей метаэвристик. Рассмотрим каждый подход подробно:

MetaheuristicEnsembleAvg – Данный класс получает на вход конструктора от одного до трех метаэвристических алгоритма оптимизации, а также опционально условия остановки их работы. Суть данного подхода заключается в том, что метаэвристики запускаются параллельно, далее после того, как все метаэвристики отработают берется среднее значение результатов их работы. Данный ансамбль метаэвристик является самым простым в реализации, а также подразумевает усреднение результата в случае, если одна из метаэвристик плохо отработала, чтобы весь результат не сильно ухудшился.

```
def solveProblem(self, problem):
    with (concurrent.futures.ThreadPoolExecutor() as executor):

        future1 = executor.submit(self.model1.solve, problem)

        if self.model2 is not None:

            future2 = executor.submit(self.model2.solve, problem)

            if self.model3 is not None:

                future3 = executor.submit(self.model3.solve, problem)

                best_position1, best_fitness1 = future1.result()
                best_position2, best_fitness2 = future2.result()
                best_position3, best_fitness3 = future3.result()
                avg_cor = self.average((best_fitness1, best_fitness2, best_fitness3))
                avg_arr = self.find_average_arrays3(best_position1, best_position2, best_position3)
                return avg_arr, avg_cor

            else:

                best_position1, best_fitness1 = future1.result()
                best_position2, best_fitness2 = future2.result()
                avg_cor = self.average((best_fitness1, best_fitness2))
                avg_arr = self.find_average_arrays2(best_position1, best_position2)
                return avg_arr, avg_cor

        else:

            best_position1, best_fitness1 = future1.result()
            return best_position1, best_fitness1
```

Рисунок 3.1 Основная реализация ансамбля MetaheuristicEnsembleAvg

На рисунке представлена основная реализация данного ансамбля метаэвристик. Он учитывает, что метаэвристик может быть как одна, так две, так и три, а далее просто усредняет результат.

MetaheuristicEnsembleLearn — Данный класс получает на вход конструктора от одного до трех метаэвристических алгоритма оптимизации, а также опционально условия остановки их работы. Суть данного подхода заключается в том, что одна метаэвристика отрабатывает и возвращает результат, далее следующая метаэвристика получает на вход результаты её работы как изначальные точки для обучения популяции и так два или три раза. Основная суть заключается в том, что у каждой метаэвристики должно быть одинаковое количество эпох жизни для того, чтобы все они работали примерно одинаковое время для повышения скорости. Обоснованием такого подхода являлся факт, что некоторые метаэвристики работают более эффективно имея начальные условия более близкие к нужному решению, чем рандомные точки.

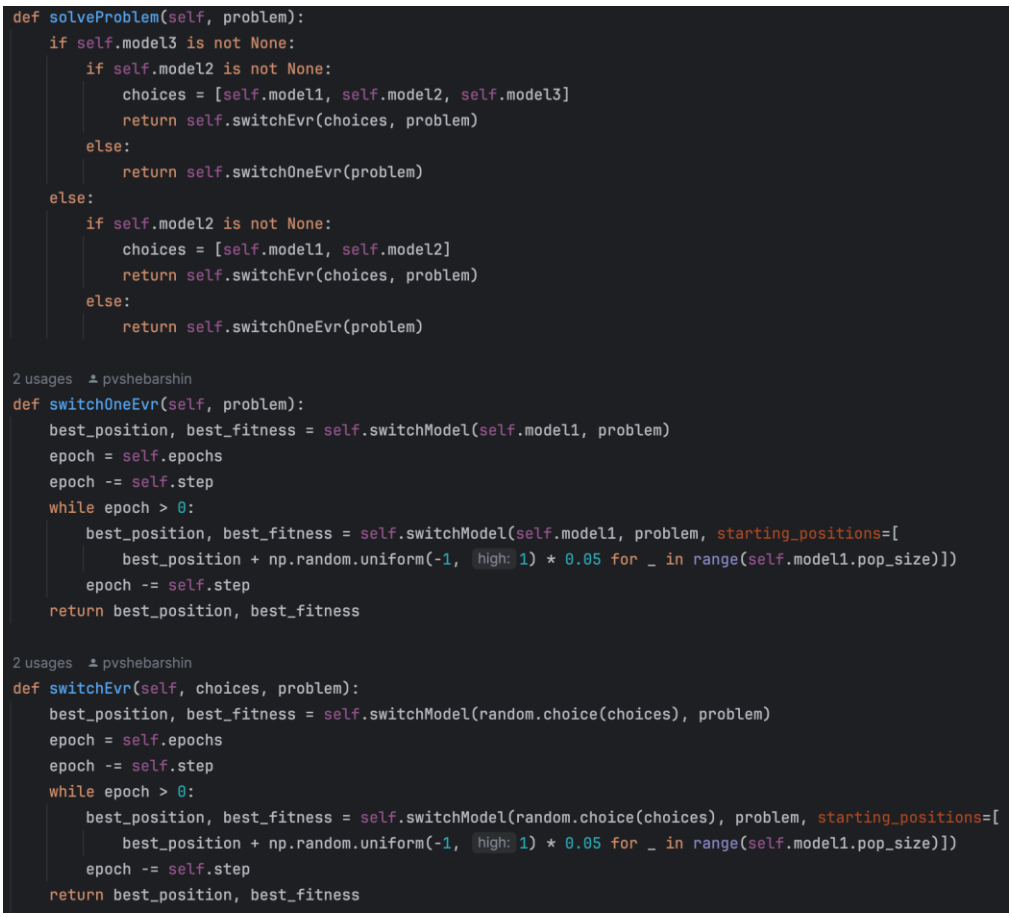
```
def solveProblem(self, problem):
    if self.model3 is not None:
        if self.model2 is not None:
            choices = [1, 2, 3]
            best_position, best_fitness = self.switchModel3(problem, choices)
            best_position, best_fitness = self.switchModel3(problem, choices, starting_positions=[
                best_position + np.random.uniform(-1, high: 1) * 0.1 for _ in range(self.model1.pop_size)])
            best_position, best_fitness = self.switchModel3(problem, choices, starting_positions=[
                best_position + np.random.uniform(-1, high: 1) * 0.01 for _ in range(self.model1.pop_size)])
            return best_position, best_fitness
        else:
            best_position, best_fitness = self.switchModel3(problem, choices: [1])
            return best_position, best_fitness
    else:
        if self.model2 is not None:
            choices = [1, 2]
            best_position, best_fitness = self.switchModel3(problem, choices)
            best_position, best_fitness = self.switchModel3(problem, choices, starting_positions=[
                best_position + np.random.uniform(-1, high: 1) * 0.1 for _ in range(self.model1.pop_size)])
            return best_position, best_fitness
        else:
            best_position, best_fitness = self.switchModel3(problem, choices: [1])
            return best_position, best_fitness
```

Рисунок 3.2 Основная реализация ансамбля MetaheuristicEnsembleLearn

На изображении показана основная реализация данного ансамбля метаэвристик. Он учитывает, что метаэвристик может быть как одна, так две, так и три, далее они отрабатывают последовательно, где каждая следующая

метаэвристика получает на вход результат работы предыдущей и в конечном итоге последняя возвращает результат работы данного ансамбля.

MetaheuristicEnsembleAlternation — Данный класс получает на вход конструктора от одного до трех метаэвристических алгоритма оптимизации, опционально условия остановки их работы, число эпох, которое в общей сложности должно быть отработано метаэвристиками, а также шаг, с которым метаэвристики должны отрабатывать эпохи. Смысл шага заключается в том, чтобы каждая метаэвристика по очереди жила столько эпох, сколько указано в шаге работы, пока число эпох не дойдет до указанного в конструкторе, по умолчанию число эпох равняется 60, а шаг 5. Основной сутью данного ансамбля является использование каждой метаэвристики несколько раз с гипотезой, что в подобном тандеме результат будет максимально оптимален, особенно с использованием метаэвристик разных типов, например взять одну роевую, одну основанную на человеческом поведении и одну основанную на использовании математических методов.



```
def solveProblem(self, problem):
    if self.model3 is not None:
        if self.model2 is not None:
            choices = [self.model1, self.model2, self.model3]
            return self.switchEvr(choices, problem)
        else:
            return self.switchOneEvr(problem)
    else:
        if self.model2 is not None:
            choices = [self.model1, self.model2]
            return self.switchEvr(choices, problem)
        else:
            return self.switchOneEvr(problem)

2 usages  pvshebarshin
def switchOneEvr(self, problem):
    best_position, best_fitness = self.switchModel(self.model1, problem)
    epoch = self.epochs
    epoch -= self.step
    while epoch > 0:
        best_position, best_fitness = self.switchModel(self.model1, problem, starting_positions=[
            best_position + np.random.uniform(-1, high: 1) * 0.05 for _ in range(self.model1.pop_size)])
        epoch -= self.step
    return best_position, best_fitness

2 usages  pvshebarshin
def switchEvr(self, choices, problem):
    best_position, best_fitness = self.switchModel(random.choice(choices), problem)
    epoch = self.epochs
    epoch -= self.step
    while epoch > 0:
        best_position, best_fitness = self.switchModel(random.choice(choices), problem, starting_positions=[
            best_position + np.random.uniform(-1, high: 1) * 0.05 for _ in range(self.model1.pop_size)])
        epoch -= self.step
    return best_position, best_fitness
```

Рисунок 3.3 Основная реализация ансамбля MetaheuristicEnsembleAlternation

На изображении показана основания реализация данного подхода на языке программирования Python. Данная реализация учитывает, что метаэвристик на входе может быть как одна, так две, так и три, а далее на выбранной проблеме каждая метаэвристика отрабатывает выбранное число эпох в рандомном порядке.

4 Результаты экспериментов и их анализ

Для тестирования и анализа работы ансамблей метаэвристик в стеганографическом алгоритме на основе дискретного преобразования Фурье были использованы тестовые изображения с сайта [28]. это База данных изображений сигналов и обработки изображений (SIPI), размещенная в Университете Южной Калифорнии (USC). База данных SIPI — это обширная коллекция изображений и видео, используемых для исследований, преподавания и разработки в области обработки сигналов и изображений. Она содержит более 10 000 изображений и видеороликов, охватывающих широкий спектр категорий, включая: естественные изображения, медицинские изображения, промышленные изображения, текстуры. База данных SIPI является ценным ресурсом для исследователей, преподавателей и разработчиков в области обработки сигналов и изображений. Ее можно использовать для обучения и тестирования алгоритмов обработки изображений, разработки новых методов обработки изображений, сравнения производительности различных алгоритмов, преподавания курсов по обработке сигналов и изображений.

Из вышеописанной базы было взято 9 изображений черно-белого спекта, а именно: area.png, boat.png, stream_and_bridge.png, airplane.png, goldhill.png, yacht.png, peppers.png, btr.png, baboon.png.

Таблица 4.1 Изображения для тестирования.

area.png

boat.png

stream_and_bridge.png

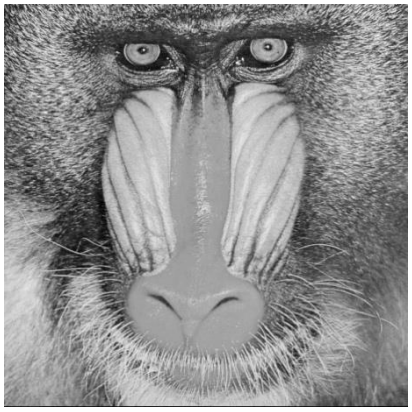
airplane.png

goldhill.png

yacht.png

peppers.png

btr.png



baboon.png

Расчёты проводились по трем ансамблям метаэвристик, в них использовалась как комбинация из двух эвристик, так и комбинации из трех различных сочетаний метаэвристик. Для усреднения времени работы при создании метаэвристик использовались следующие критерии выбора эпох популяций. Для ансамбля с комбинаций из трех эвристик для каждой использовалось по 20 эпох популяции, для ансамбля с комбинацией из двух эвристик использовалось по 30 эпох, а для проверок на одной эвристике использовалось 60 эпох. Таким образом для всех вариаций работы эвристик на одной задаче было использовано 60 эпох популяции. Однако исключением был ансамбль MetaheuristicEnsembleAlternation, так как он чередует эвристики по 5 эпох вне зависимости от их количеств, но в конце концов пройдет все равно 60 эпох, так как по настройкам его работы должно было пройти 12 итераций для этого. Для встраивания были выбраны как осмысленные сообщения (для PSNR), так и рандомная последовательность

бит (для емкости), для каждой пары изображение/ансамбль было произведено по 10 измерений, измерения проводились на MacBook Air m2 512Гб 16Гб (2022 года).

Для анализа работы ансамблей метаэвристик было выбрано две основные метрики — PSNR [26] и емкость изображения. PSNR (Peak Signal-to-Noise Ratio) — это метрика, используемая для измерения качества восстановленного изображения по сравнению с оригинальным изображением. PSNR часто применяется в области обработки изображений и видео, а также в сфере сжатия данных. Для использования PSNR важно понимать несколько ключевых понятий:

1. Сигнал — это представление оригинального изображения или видео без потерь.

2. Шум — разница между оригинальным сигналом и восстановленным сигналом. Чем меньше шума, тем качественнее восстановленное изображение.

3. Максимальное значение яркости — это максимально возможное значение яркости для пикселя в изображении. Обычно это 255 для изображений в формате 8 бит на канал.

PSNR вычисляется с использованием формулы:

$$PSNR = 10 * \log_{10}(\frac{MAX^2}{MSE})$$

где: MAX - максимальное значение яркости (обычно 255), MSE (Mean Squared Error) - среднеквадратичная ошибка между оригинальным и восстановленным изображениями.

Чем выше значение PSNR, тем лучше качество восстановленного изображения. Высокое значение PSNR указывает на то, что разница между оригинальным и восстановленным изображениями незначительна, и изображение сохраняет высокую степень точности. PSNR широко используется для сравнения качества изображений после их сжатия или обработки, так как позволяет количественно оценить степень потерь

информации при таких операциях. Однако PSNR не всегда является идеальной метрикой, так как не всегда коррелирует с визуальным восприятием человека. Например, высокое значение PSNR не гарантирует, что изображение будет выглядеть хорошо для человеческого глаза.

Емкость цифрового изображения, обработанного стеганографическим алгоритмом, определяет скрытую информацию, которая была внедрена в изображение. Емкость цифрового изображения, обработанного стеганографическим алгоритмом, зависит от различных факторов, включая:

1. Алгоритм стеганографии: разные алгоритмы могут иметь различную емкость и способы встраивания информации.

2. Разрешение и глубина цвета изображения: чем выше разрешение и глубина цвета, тем больше емкость может быть использована для скрытия информации.

3. Степень сжатия изображения: сжатие изображения может повлиять на возможность эффективного скрытия информации.

4. Уровень защиты информации: для надежной и безопасной передачи скрытой информации может потребоваться использование более сложных алгоритмов с меньшей емкостью.

Из этого следует, что емкость цифрового изображения после обработки стеганографическим алгоритмом является ключевым показателем для определения количества скрытой информации, которая может быть передана или хранится в изображении без заметного ухудшения его качества или безопасности. В данной работе емкость изображения измеряется в битах, которые можно записать в изображение на пиксель этого изображения.

Для начала стоит рассмотреть усредненные результаты по всем картинкам и всем ансамблям.

Таблица 4.2 Усредненные результаты работы ансамбля MetaheuristicEnsembleAvg по каждому изображению для комбинации из двух метаэвристик.

Имя файла	Время (с)	PSNR (Дб)	Ёмкость (б/п)
airplane.png	90.4819	50.1027	0.2389
area.png	90.2205	30.4069	0.2281
baboon.png	90.4641	39.7004	0.1182
boat.png	88.9099	52.4293	0.2134
btr.png	90.6596	45.1185	0.1473
goldhill.png	91.0382	41.6215	0.2054
peppers.png	90.0121	49.4534	0.2586
stream_and_bridge.png	89.2930	41.2083	0.1089
yacht.png	91.0761	46.3077	0.1543

Таблица 4.3 Усредненные результаты работы ансамбля MetaheuristicEnsembleLearn по каждому изображению для комбинации из двух метаэвристик.

Имя файла	Время (с)	PSNR (Дб)	Ёмкость (б/п)
airplane.png	57.5548	51.1907	0.2401
area.png	56.5466	30.5315	0.2310
baboon.png	57.0877	40.7841	0.1211
boat.png	57.5302	53.4197	0.2172
btr.png	57.5876	45.9014	0.1501
goldhill.png	58.3563	41.8402	0.2091
peppers.png	58.1822	51.0921	0.2616
stream_and_bridge.png	56.94	42.1368	0.1112
yacht.png	57.986	48.0782	0.1571

Таблица 4.4 Усредненные результаты работы ансамбля MetaheuristicEnsembleAlternation по каждому изображению для комбинации из двух метаэвристик.

Имя файла	Время (с)	PSNR (Дб)	Ёмкость (б/п)
airplane.png	65.2371	50.9891	0.2423
area.png	63.286	30.5098	0.2332
baboon.png	64.3451	40.4541	0.1237

boat.png	64.9358	53.0671	0.2196
btr.png	64.5124	45.6973	0.1544
goldhill.png	65.6669	41.8947	0.2129
peppers.png	64.9244	50.7162	0.2634
stream_and_bridge.png	63.8322	41.975	0.1155
yacht.png	64.6301	47.7509	0.1695

Таблица 4.5 Усредненные результаты работы ансамбля MetaheuristicEnsembleAvg по каждому изображению для комбинации из трех метаэвристик.

Имя файла	Время (с)	PSNR (Дб)	Ёмкость (б/п)
airplane.png	105.9669	49.6990	0.2401
area.png	105.0157	30.3690	0.2291
baboon.png	106.3888	39.4651	0.1202
boat.png	105.1728	52.4185	0.2191
btr.png	106.4778	44.9737	0.1513
goldhill.png	106.3657	41.4082	0.2123
peppers.png	106.8252	49.2781	0.2622
stream_and_bridge.png	105.5913	41.0567	0.1119
yacht.png	105.7593	46.2755	0.1591

Таблица 4.6 Усредненные результаты работы ансамбля MetaheuristicEnsembleLearn по каждому изображению для комбинации из трех метаэвристик.

Имя файла	Время (с)	PSNR (Дб)	Ёмкость (б/п)
airplane.png	57.4794	51.5708	0.2431
area.png	56.1912	30.7056	0.2311
baboon.png	56.5529	41.1043	0.1312
boat.png	56.8995	53.4668	0.2242
btr.png	57.1796	45.9121	0.1591
goldhill.png	58.5767	42.0601	0.2194
peppers.png	57.3861	51.0159	0.2696

stream_and_bridge.png	55.9708	42.2604	0.1199
yacht.png	57.6418	48.2543	0.1612

Таблица 4.7 Усредненные результаты работы ансамбля MetaheuristicEnsembleAlternation по каждому изображению для комбинации из трех метаэвристик.

Имя файла	Время (с)	PSNR (Дб)	Ёмкость (б/п)
airplane.png	64.2591	51.3499	0.2494
area.png	63.0743	30.5995	0.2393
baboon.png	63.8366	40.6551	0.1395
boat.png	64.1271	53.1505	0.2301
btr.png	64.3796	45.7058	0.1653
goldhill.png	65.3660	41.8729	0.2264
peppers.png	64.4999	50.9489	0.2777
stream_and_bridge.png	63.2949	42.0251	0.1262
yacht.png	64.3167	47.9060	0.1655

Также для анализа данных результатов работы ансамблей стоит взглянуть на результаты работы одиночных метаэвристик.

Таблица 4.8 Усредненные результаты работы по каждому изображению для одиночных метаэвристик.

Имя файла	Время (с)	PSNR (Дб)	Ёмкость (б/п)
airplane.png	65.4145	50.7557	0.2421
area.png	63.6605	30.5048	0.2299
baboon.png	65.0807	40.6547	0.1201
boat.png	65.3435	53.0119	0.2165
btr.png	65.4439	45.3378	0.1488
goldhill.png	65.7229	41.9147	0.2091
peppers.png	65.3679	50.4196	0.2616
stream_and_bridge.png	64.5545	41.8088	0.1111
yacht.png	64.8779	47.3599	0.1572

Как видно из полученных измерений MetaheuristicEnsembleAvg показывает худшие результаты по скорости, это связано со спецификой

создания потоков в языке программирования Python, получается так, что при анализе каждого блока изображения создается новый поток по несколько раз, особенно это заметно учитывая факт того, что стеганографический алгоритм повторяет итерации в случае ошибок извлечения, исходя из этого можно сделать вывод, что его использование не является целесообразным в случае если важна скорость исполнения программы. MetaheuristicEnsembleAlternation и MetaheuristicEnsembleLearn показывают более высокие результаты в сравнении с MetaheuristicEnsembleAvg как по времени, так и по PSNR с ёмкостью изображения, однако пусть и не сильно, но MetaheuristicEnsembleLearn во всем немного превосходит MetaheuristicEnsembleAlternation. Для подтверждения этого стоит взглянуть на усредненные показатели по ансамблям.

Теперь стоит взглянуть на средние результаты метрик по ансамблям.

Таблица 4.9 Усредненные результаты работы по каждому ансамблю для комбинации из двух метаэвристик.

Имя файла	Время (с)	PSNR (Дб)	Ёмкость (б/п)
MetaheuristicEnsembleAlternation	64.5967	44.7838	0.1995
MetaheuristicEnsembleAvg	90.2395	44.0387	0.1743
MetaheuristicEnsembleLearn	57.5301	44.9972	0.2054

Таблица 4.10 Усредненные результаты работы по каждому ансамблю для комбинации из трех метаэвристик.

Имя файла	Время (с)	PSNR (Дб)	Ёмкость (б/п)
MetaheuristicEnsembleAlternation	64.1282	44.9126	0.2014
MetaheuristicEnsembleAvg	105.951	43.8826	0.1819
MetaheuristicEnsembleLearn	57.0975	45.1500	0.2136

Данные таблицы только подтверждают факт, что MetaheuristicEnsembleLearn является лучшим ансамблем из рассмотренных в работе, также видно, что он превосходит работу метаэвристик по отдельности, далее, стоит рассмотреть какие комбинации метаэвристик

показали самый наилучший результат в работе этого ансамбля и для контраста увидеть худшие.

Таблица 4.11 Наилучшие по работе метаэвристики в комбинации из двух для ансамбля MetaheuristicEnsembleLearn.

Эвристики	Время (с)	PSNR (Дб)	Ёмкость (б/п)
DE_HHO	27.9683	46.0983	0.2102
PSO_HHO	32.9882	46.0417	0.2088
HHO_BBO	32.9193	46.0340	0.2083
HHO_ACO	45.7494	45.9487	0.2077
DE_BBO	32.6803	45.8463	0.207
HHO_IWO	70.4511	45.8417	0.2061
HHO_CA	25.6366	45.4918	0.2055

Таблица 4.12 Наихудшие по работе метаэвристики в комбинации из двух для ансамбля MetaheuristicEnsembleLearn.

Эвристики	Время (с)	PSNR (Дб)	Ёмкость (б/п)
GBO_ACO	68.4542	43.8506	0.192
GBO_ABC	60.2782	43.8936	0.1983
PSO_ACO	56.9241	44.1479	0.1972
PSO_ABC	49.4700	44.1844	0.1977
FBIO_GBO	85.2945	44.2854	0.1954
GBO_CA	39.1777	44.3092	0.1961
ABC_CA	45.6334	44.3530	0.1951

Таблица 4.13 Наилучшие по работе метаэвристики в комбинации из трех для ансамбля MetaheuristicEnsembleLearn.

Эвристики	Время (с)	PSNR (Дб)	Ёмкость (б/п)
DE_HHO_FBIO	60.2096	46.2135	0.2166
HHO_GBO_IWO	73.7233	46.2076	0.2155
HHO_ABC_BBO	40.6803	46.2067	0.2151
FBIO_IWO_BBO	77.3914	46.1390	0.2149
PSO_HHO_BBO	33.0652	46.1389	0.2141

DE_HHO_ABC	42.0760	46.1316	0.2138
HHO_FBIO_IWO	86.7455	46.0085	0.2136

Таблица 4.13 Наихудшие по работе метаэвристики в комбинации из трех для ансамбля MetaheuristicEnsembleLearn.

Эвристики	Время (с)	PSNR (Дб)	Ёмкость (б/п)
DE_FBIO_GBO	65.7149	43.7223	0.201
PSO_FBIO_GBO	67.8830	43.9210	0.2012
PSO_GBO_CA	34.4099	44.0262	0.2015
DE_PSO_GBO	40.6153	44.0428	0.2
GBO_ABC_ACO	70.1491	44.0742	0.2043
DE_CA_ACO	51.4519	44.1027	0.2088
PSO_FBIO_ACO	74.2893	44.1086	0.1989

Как видно из таблиц наилучшие сочетания из эвристик — это сочетания DE и HHO для комбинации из двух, а в комбинации из трех к ним добавляется ещё FBIO. В принципе во всех лучших результатах фигурирует HHO, это показывает, что Harris Hawks Optimization является весьма многозадачной эвристикой, которая подходит для решения многих задач, она также показала хорошие результаты на тестовых функциях. При этом видно, что даже худшие сочетания метаэвристик показали весьма неплохие результаты в абсолютных значнеиях.

Исходя из просмотренных таблиц с результатами, для решения задачи оптимизации сокрытия данных алгоритмом на основе дискретного преобразования Фурье наилучшим решением можно назвать ансамбль MetaheuristicEnsembleLearn с комбинацией эвристик Harris Hawks Optimization и Differential Evolution так как они работают на 53% быстрее чем лучший вариант из трех эвристик, а ухудшение работы по метрикам при этом меньше, чем 1%.

Данная тема имеет большой простор для дальнейших исследований, однако исходя из проведенных опытов в данной работе можно сделать

вывод, что наиболее перспективным вектором исследования является подход в котором одна метаэвристика получает на вход обученную популяцию другой эвристики, этот подход может нивелировать недостатки определенной эвристики и повысить эффективность оптимизации взяв плюсы от комбинации различных подходов к оптимизации. Реализацию ансамблей и расчеты можно найти в репозитории [29].

5 Заключение

В рамках данной работы, после анализа научно-технической литературы были реализованы метаэвристики Artificial Bee Colony (ABC), Biogeography-Based Optimization (BBO), Culture Algorithm (CA), Ant Colony Optimization Continuous (ACO), Harris Hawks Optimization (HHO) и Invasive Weed Optimization (IWO), Differential Evolution (DE), Forensic-Based Investigation Optimization (FBIO), Gradient-Based Optimizer (GBO), Particle Swarm Optimization (PSO), для которых были разработаны и реализованы три ансамбля оптимизации: MetaheuristicEnsembleAvg, MetaheuristicEnsembleLearn, MetaheuristicEnsembleAlternation. Метаэвристики были протестированы на тестовых функциях, после чего был реализован на языке программирования Python алгоритм встраивания информации в частотную область дискретного преобразования Фурье цифровых изображений [26]. Далее по целевой функции, которая анализировала область встраивания данных, была проведена оптимизация, которая в дальнейшем была проанализирована по времени, PSNR и емкости. На основе полученных результатов был сделан вывод, что наилучший вариант для оптимизации из рассмотренных — это ансамбль MetaheuristicEnsembleLearn с комбинацией эвристик HHO и DE.

6 Список использованных источников

[1] R. Abbassi, A. Abbassi, A.A. Heidari, S. Mirjalili, An efficient salp swarm-inspired algorithm for parameters identification of photovoltaic cell models, Energy Convers. Manage. 179 (2019) 362–372.

[2] H. Faris, A.M. Al-Zoubi, A.A. Heidari, I. Aljarah, M. Mafarja, M.A. Hassonah, H. Fujita, An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks, Inf. Fusion 48 (2019) 67–83.

[3] J. Nocedal, S.J. Wright, Numerical Optimization, 2nd ed., 2006.

[4] G. Wu, Across neighborhood search for numerical optimization Inform. Sci. 329 (2016) 597–618.

[5] G. Wu, W. Pedrycz, P.N. Suganthan, R. Mallipeddi, A variable reduction strategy for evolutionary algorithms handling equality constraints, Appl. Soft Comput. 37 (2015) 774–786.

[6] J. Dr éo, A.P. étrowski, P. Siarry, E. Taillard, Metaheuristics for Hard Optimization: Methods and Case Studies, Springer Science & Business Media, 2006.

[7] E.-G. Talbi, Metaheuristics: from Design to Implementation, vol. 74, John Wiley & Sons, 2009.

[8] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680.

[9] J.H. Holland, Genetic algorithms, Sci. Am. 267 (1992) 66–73.

[10] J. Luo, H. Chen, Y. Xu, H. Huang, X. Zhao, et al., An improved grasshopper optimization algorithm with application to financial stress prediction, Appl. Math. Model. 64 (2018) 654–668.

[11] M. Wang, H. Chen, B. Yang, X. Zhao, L. Hu, Z. Cai, H. Huang, C. Tong, Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses, Neurocomputing 267 (2017) 69–84.

[12] L. Shen, H. Chen, Z. Yu, W. Kang, B. Zhang, H. Li, B. Yang, D. Liu, Evolving support vector machines using fruit fly optimization for medical data classification, Knowl.-Based Syst. 96 (2016) 61–75.

[13] Q. Zhang, H. Chen, J. Luo, Y. Xu, C. Wu, C. Li, Chaos enhanced bacterial foraging optimization for global optimization, IEEE Access (2018).

[14] A.A. Heidari, R.A. Abbaspour, A.R. Jordehi, An efficient chaotic water cycle algorithm for optimization tasks, Neural Comput. Appl. 28 (2017) 57–85.

[15] M. Mafarja, I. Aljarah, A.A. Heidari, A.I. Hammouri, H. Faris, M. A.-Z. Ala', S. Mirjalili, Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems, Knowl.-Based Syst. 145 (2018a) 25–45.

[16] Dan Simon Biogeography-Based Optimization // IEEE. - Cleveland: IEEE, 2008. - C. 702 - 713.

[17] A.R. Mehrabian, C. Lucas, A novel numerical optimization algorithm inspired from weed colonization, Ecological Informatics, Volume 1, Issue 4, 2006, 355-366

[18] Ali W. Mohamed, Anas A. Hadi, Kamal M. Jambi, Novel mutation strategy for enhancing SHADE and LSHADE algorithms for global numerical optimization, Swarm and Evolutionary Computation, Volume 50, 2019

[19] Chen, B., Zhao, L. and Lu, J.H., 2009, April. Wind power forecast using RBF network and culture algorithm. In 2009 International Conference on Sustainable Power Generation and Supply (pp. 1-6). IEEE.

[20] Jui-Sheng Chou, Ngoc-Mai Nguyen, FBI inspired meta-optimization, Applied Soft Computing, Volume 93, 2020, 106339

[21] Ahmadianfar, I., Bozorg-Haddad, O. and Chu, X., 2020. Gradient-based optimizer: A new metaheuristic optimization algorithm. Information Sciences, 540, 131-159.

[22] B. Basturk, D. Karaboga, An artificial bee colony (ABC) algorithm for numeric function optimization, in: IEEE Swarm Intelligence Symposium 2006, May 12–14, Indianapolis, IN, USA, 2006.

[23] Socha, K. and Dorigo, M., 2008. Ant colony optimization for continuous domains. European journal of operational research, 185(3), 1155-1173.

[24] Heidari, A.A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M. and Chen, H., 2019. Harris hawks optimization: Algorithm and applications. Future generation computer systems, 97, 849-872.

[25] Kennedy, J. and Eberhart, R., 1995, November. Particle swarm optimization. In Proceedings of ICNN'95-international conference on neural networks (Vol. 4, pp. 1942-1948). IEEE.

[26] Anna Melman, Oleg Evsutin Comparative study of metaheuristic optimization algorithms for image steganography based on discrete Fourier transform domain // Soft Computing. - Moscow: ELSEVIER, 2023. - С. 1 - 21.

[27] Comparative evaluation of the effectiveness of metaheuristic optimization algorithms // GitHub URL: https://github.com/pvshebarshin/Comparative-evaluation-of-the-effectiveness-of-metaheuristic-optimization-algorithms

[28] Signal and Image Processing Institute // USC URL: https://sipi.usc.edu/database/database.php?volume=misc (дата обращения: 01.03.2024).

[29] Watermark Embedder // GitHub URL: https://github.com/pvshebarshin/watermark-embedder