МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Львівський національний університет імені Івана Франка
Факультет електроніки та комп'ютерних технологій
Кафедра радіоелектронних і комп’ютерних систем

Допустити до захисту

Завідувач кафедри
_____ доцент Оленич І. Б.
«____»_____ 2023 р.

Кваліфікаційна робота

Магістр
(освітній ступінь)

Розробка системи візуалізації алгоритму штучного бджолиного рою.

Виконала:
Студентка V курсу групи ФЕС-54
спеціальності 121 – Комп’ютерні науки

_____ Суходольська І.І.

Науковий керівник:
_____ Сінкевич О. О.
«____»_____ 2023 р.

Львів — 2023

ЗМІСТ

ВСТУП ……4

РОЗДІЛ 1 Матаеврестичні алгоритми……..8

1.1. Метаевристики ітераційного типу……..8

1.2. Метаевристики на основі стохастичного локального пошуку……8

1.3. Еволюційні Метаевристики……..9

1.4. Метаевристики на основі ройового інтелекту……9

РОЗДІЛ 2 Ройовий інтелект……..10

2.1. Основні принципи колективного інтелекту……..10

2.1.1. Самоорганізація ……..11

2.1.2. Координація……..11

2.1.3. Спеціалізація……11

2.2. Особливості ройового інтелекту……12

2.3. Сфери застосування ройового інтелекту……13

РОЗДІЛ 3 Бджолиної алгоритм ……14

3.1.Штучний бджолиний рій……14

3.2. Алгоритм штучного бджолиного рою (ABC)……..15

3.3. Особливості алгоритму штучної бджолиної колонії……16

3.4. Застосування ABC……16

3.5. принцип роботи ABC……17

3.6. Етапи алгоритму штучного бджолиного рою……20

3.6.1. Етап 1……...20

3.6.2. Етап 2 ……20

3.6.3. Етап 3……20

РОЗДІЛ 4 Основні особливості алгоритму штучного бджолиного рою……..22

РОЗДІЛ 5 Реалізація ABC…….23

ВИСНОВКИ ……..25

ВИКОРИСТАНІ ДЖЕРЕЛА…….26

ВСТУП

Штучний інтелект можна визначити як моделювання розумної поведінки в природі. Однією з технік, в яких спостерігається штучний інтелект, є метаевристичні алгоритми. Вони успішно використовуються при вирішенні багатьох завдань з різними характеристиками. Успішні результати підвищили інтерес до метаевристичних алгоритмів. Відомо, що сьогодні існує більше двохсот метаевристичних алгоритмів. Птахи, комахи, тварини, людина, рослина, вода, екосистема, електромагнітна сила, гравітація, природна еволюція є джерелами натхнення для розробки метаевристичних алгоритмів. Метаевристичні алгоритми класифікуються за джерелом натхнення. Це еволюційні алгоритми, алгоритми на основі фізики, алгоритми на основі роїв, алгоритми, натхненні біологією, і алгоритми, натхненні природою. Алгоритми на основі роїв займають важливе місце в літературі. Ці алгоритми моделюють інтелектуальні особливості поведінки рою. Такі особливості, як позитивний зворотний зв’язок, негативний зворотний зв’язок, флуктуації та численні взаємодії алгоритмів на основі роїв забезпечують гнучкість і універсальність

В даній курсовій роботі ми розглянемо візуалізацію алгоритму штучного бджолиного рою. Адже, візуалізація алгоритму штучного бджолиного рою має кілька важливих причин і застосувань у сучасному контексті. Основні з них включають:

Розуміння та аналіз алгоритму: Візуалізація надає можливість краще розуміти та аналізувати роботу алгоритму штучного бджолиного рою. Вона дозволяє візуально спостерігати за взаємодією бджіл та їхніми діями, рухами та обміном інформацією. Це допомагає дослідникам та розробникам краще зрозуміти поведінку алгоритму та його вплив на пошук оптимального рішення.

Візуалізація процесу оптимізації: Візуалізація дозволяє відображати процес пошуку оптимального рішення на просторі параметрів. Це може бути корисним

для відстеження еволюції рішень, виявлення трендів та паттернів, а також виявлення проблем або недоліків алгоритму. Візуальне представлення допомагає зрозуміти, як розвиваються рішення з часом та які фактори впливають на їхню якість.

Навчання та освіта: Візуалізація алгоритму штучного бджолиного рою може бути використана для навчання та освіти. Вона допомагає студентам та новачкам у галузі інформатики та оптимізації краще розуміти та візуалізувати принципи роботи алгоритму, його етапи та взаємодію агентів. Візуалізація сприяє покращенню навчального процесу та полегшує засвоєння складних концепцій.

Подача результатів та комунікація: Візуалізація є потужним інструментом для подачі результатів досліджень та комунікації зі спільнотою. Вона дозволяє ефективно відображати результати роботи алгоритму, порівнювати різні варіанти та показувати вплив різних параметрів на процес оптимізації. Візуалізація допомагає зробити презентації та публікації більш доступними й зрозумілими для аудиторії.

Все це підкреслює важливість візуалізації алгоритму штучного бджолиного рою для розуміння, аналізу, навчання та комунікації в цій сфері.

INTRODUCTION

Artificial intelligence can be defined as the modeling of intelligent behavior in nature. One of the techniques where artificial intelligence is observed is metaheuristic algorithms. They are successfully used in solving various tasks with different characteristics. The successful results have increased the interest in metaheuristic algorithms. It is known that there are over two hundred metaheuristic algorithms today. Birds, insects, animals, humans, plants, water, ecosystems, electromagnetic force, gravity, natural evolution serve as sources of inspiration for the development of metaheuristic algorithms. Metaheuristic algorithms are classified based on the source

of inspiration. These include evolutionary algorithms, physics-based algorithms, swarm-based algorithms, biology-inspired algorithms, and nature-inspired algorithms. Swarm-based algorithms hold an important place in the literature. These algorithms model the intelligent features of swarm behavior. Features such as positive feedback, negative feedback, fluctuations, and numerous interactions provide flexibility and universality.

In this research paper, we will explore the visualization of the artificial bee colony algorithm. Visualization of the artificial bee colony algorithm has several important reasons and applications in the modern context. The main ones include:

Understanding and analyzing the algorithm: Visualization provides an opportunity to better understand and analyze the workings of the artificial bee colony algorithm. It allows for visually observing the interaction of bees and their actions, movements, and information exchange. This helps researchers and developers better understand the behavior of the algorithm and its impact on finding optimal solutions.

Visualization of the optimization process: Visualization allows for displaying the process of searching for an optimal solution in parameter space. This can be useful for tracking the evolution of solutions, identifying trends and patterns, as well as identifying issues or shortcomings in the algorithm. Visual representation helps understand how solutions evolve over time and what factors influence their quality.

Learning and education: Visualization of the artificial bee colony algorithm can be used for learning and education purposes. It helps students and newcomers in the field of computer science and optimization to better understand and visualize the principles of the algorithm, its stages, and the interaction of agents. Visualization contributes to improving the learning process and facilitates the assimilation of complex concepts.

Results presentation and communication: Visualization is a powerful tool for presenting research results and communicating with the community. It allows for

effectively showcasing the results of the algorithm's work, comparing different variants, and demonstrating the impact of different parameters on the optimization process. Visualization helps make presentations and publications more accessible and understandable to the audience.

All of this emphasizes the importance of visualizing the artificial bee colony algorithm for understanding, analysis, learning, and communication in this field.

РОЗДІЛ 1 Матаеврестичні алгоритми

Метаевристики класифікують відповідно до парадигми, яка використовується при їх побудові. Буде розглянуто підхід до класифікації та наведені приклади поширених метаевристик, серед яких такі:

1.1. Матаевристики ітераційного типу

1. Метаевристики ітераційного типу - це клас оптимізаційних алгоритмів, які працюють на основі ітераційного процесу, повторюючи цикли покращення поточного рішення. Вони поступово збільшують якість рішення, застосовуючи пошук у просторі рішень з використанням метаевристичних принципів, таких як просування в напрямку кращих розв'язків, випадковий пошук, або експлуатація знань про проблему. Ці алгоритми широко використовуються для розв'язання складних оптимізаційних завдань у різних галузях, таких як логістика, розкладання задач, оптимізація маршрутів та управління виробництвом.

Виділяють такі поширені Метаевристики цього типу:

- пошук зі змінюваними (пульсуючими) околами;
- керований локальний пошук;
- табу-пошук.

1.2. Метаевристики на основі стохастичного локального пошуку

2. Метаевристики на основі стохастичного локального пошуку.

Використовують у якості підлеглої процедуру локального пошуку, у яку вбудовано ймовірнісні механізми. Це, зокрема, дозволяє будувати нерелаксаційні обчислювальні схеми з метою уникнення передчасної збіжності.

Найбільш відомі такі алгоритми:

- метод звужувальних околів;
- алгоритми імітаційного відпалу;
- О-алгоритми;

- повторюваний локальний пошук;

- квантовий відпал;

- метод кросс-ентропії;

- GRASP.

1.3. Еволюційні Метаевристики

3. Еволюційні Метаевристики - це алгоритми пошуку, що використовуються для розв'язання задач оптимізації і моделювання шляхом послідовного відбору, комбінування і варіації шуканих параметрів розв'язків із використанням механізмів, що нагадують біологічну еволюцію.

Найбільш відомими є:

- генетичні алгоритми;

- міметичні алгоритми;

- імунні алгоритми. Варто відзначити, що за певної інтерпретації складових компонент до цього класу можна віднести і ряд метаевристик, які описуються авторами у інших термінах.

1.4. Метаевристики на основі ройового інтелекту

4. Метаевристики на основі ройового інтелекту - децентралізовані системи простих агентів, які локально взаємодіють як з середовищем, так і між собою. Часто їх об'єднують під назвою "навіяні природою чи засновані на принципах біології". Ці Метаевристики бурхливо розвиваються в останні роки, тому важко скласти повний перелік; відмітимо ті, які вже пройшли певний період досліджень і апробації:

- мурашині алгоритми;

- оптимізація роєм часток;

- бджолині алгоритми;

- алгоритми світлячків;

- алгоритми моделювання поведінки бактерій.

Останнім часом запропоновані і досліджуються також алгоритми комбінаторної оптимізації, засновані на моделюванні чи імітації поведінки косяків риб, жаб, летючих миш.

РОЗДІЛ 2 Ройовий інтелект(swarm ai)

Ройовий інтелект (або англ. swarm intelligence) відноситься до галузі штучного інтелекту, яка моделює поведінку колективів простих агентів з метою розв'язання складних завдань. Ця концепція бере натхнення зі спостережень над природними системами, такими як мурашині колонії, рої бджіл, зграї птахів і риби. Ройовий інтелект показує, що навіть незначні агенти з простими правилами взаємодії можуть виявляти вражаючу колективну поведінку та здатність до самоорганізації.

Ройовий інтелект має потужний потенціал у багатьох інших сферах, таких як фінанси, медицина, розумні міста, агротехнології та багато інших. Його здатність до колективного інтелектуального розв'язання проблем робить його цікавим інструментом для ефективного вирішення складних задач у різних галузях.

Системи ройового інтелекту, як правило, складаються з множини агентів (мультиагентна система), які локально взаємодіють між собою, а також з навколишнім середовищем. Самі агенти зазвичай досить прості, але всі разом, взаємодіючи, вони створюють, так званий, ройовий інтелект. Прикладом у природі може служити колонія мурах, рій бджіл, зграя птахів, косяк риб.

2.1. Основні принципи колективного інтелекту

Для визначення основних принципів колективного інтелекту досить розглянути принципи поведінки різних колективних комах. До колективних комах відносяться комахи, які живуть колоніями, наприклад, мурахи, бджоли,

терміти, деякі види ос і т.п. В основі поведінки колоній таких комах лежить самоорганізація.

2.1.1. Самоорганізація

Самоорганізація – множина динамічних механізмів, відповідно до яких система регулюється на глобальному рівні за рахунок взаємодії її компонентів на нижньому рівні без прямої взаємодії між цими компонентами.

Можна виділити чотири функції, які виконують соціальні комахи в процесі вирішення різних задач: координація, кооперація, колективне прийняття рішень, спеціалізація. Вони є не взаємно виключними, а навпаки – спільно вносять вклад у досягнення поставленою перед колонією цілі.

2.1.2. Координація

Координація – відповідна організація задач окремих індивідів у часі й просторі, що дозволяє вирішити виниклу перед колонією проблему. Кооперація досягається за рахунок того, що особини виконують разом загальну задачу, що не могла б бути вирішена окремим індивідом. Колективне прийняття рішень відноситься до механізмів, які спрацьовують, коли колонія зіштовхується із проблемою вибору. Цей механізм завершується колективним вибором одного з можливих рішень.

2.1.3. Спеціалізація

Спеціалізація полягає в тому, що різні дії виконуються окремими спеціалізованими групами індивідів.

Алгоритм колонії мурах (мурашиний алгоритм, алгоритм оптимізації мурашиної колонії, англ. ant colony optimization, ACO) – один з ефективних поліноміальних алгоритмів для знаходження наближених розв‘язків задачі комівояжера, а також завдань пошуку маршрутів на графах. Алгоритм можна описати у 3 етапи (рис. 1):

Перша мураха знаходить джерело їжі (F) через якийсь шлях (a). Потім повертається до гнізда (N), залишивши за собою слід з феромонів (b).

Мурахи вибирають будь-який шлях, але шлях, на якому міститься феромон робить його більш привабливішим як найкоротший шлях.

Мурахи вибирають коротший шлях, а з часом інші шляхи втрачають щільність феромонового сліду.

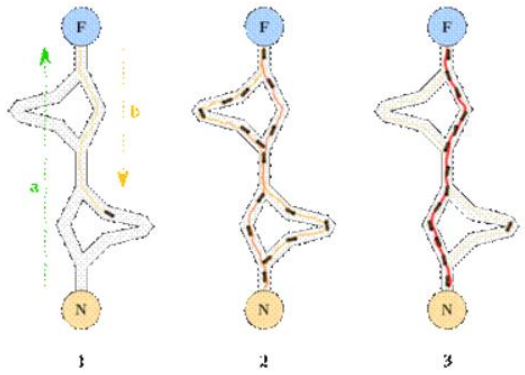

Рис. 1 – Поведінка мурах в колонії

Алгоритм колонії мурах може бути успішно застосований для вирішення складних комплексних завдань оптимізації. Мета вирішення складних комплексних завдань оптимізації - пошук і визначення найбільш відповідного рішення для оптимізації (знаходження мінімуму або максимуму) цільової функції (ціни, точності, часу, відстані тощо) з дискретної множини можливих рішень.

2.2. Особливості ройового інтелекту

Особливості ройового інтелекту:

1. Децентралізація: Ройовий інтелект базується на децентралізованій структурі, де кожен агент діє незалежно від інших. Відсутність глобального контролю дозволяє системі бути гнучкою та стійкою до випадкових впливів і збоїв.

2. Взаємодія та комунікація: Агенти взаємодіють між собою, обмінюючись інформацією та сприймаючи оточуюче середовище. Це дозволяє системі здійснювати колективну координацію та вирішувати задачі, які вимагають групової співпраці.

3. Самоорганізація: Ройовий інтелект виявляє властивість самоорганізації, де колективне поведінка виникає без централізованого керівництва або планування. Агенти взаємодіють локально за допомогою простих правил, і це призводить до глобальної поведінки системи.

4. Адаптивність: Ройовий інтелект може адаптуватися до змін в середовищі, швидко реагувати на нові умови і знаходити оптимальні рішення в реальному часі.

2.3. Сфери застосування ройового інтелекту

Сфери застосування ройового інтелекту:

1. Оптимізація маршрутів: Ройовий інтелект може бути використаний для пошуку найкоротшого маршруту в транспортних системах, логістиці та розподільчих мережах. Агенти можуть кооперуватися та обмінюватись інформацією, щоб забезпечити ефективну організацію руху та зменшити загальний час подорожі.

2. Оптимізація виробничих процесів: Ройовий інтелект може бути застосований у виробничих системах для оптимізації розкладів виробництва, розподілу завдань між робочими машинами, управління запасами та розміщення обладнання. Він дозволяє забезпечити ефективне використання ресурсів та мінімізувати затрати.

3. Колективне розпізнавання та класифікація: Ройовий інтелект може бути використаний для розпізнавання образів, класифікації даних та вирішення завдань штучного зору. Агенти можуть працювати разом, співпрацюючи при аналізі складних зразків або прийнятті колективних рішень на основі спільної інформації.

4. Оптимізація соціальних мереж: Ройовий інтелект може бути застосований у соціальних мережах для покращення рекомендаційних систем, виявлення спільнот та аналізу потоків даних. Він може допомогти виявляти нові тренди, забезпечити персоналізовані рекомендації та покращити користувацький досвід.

РОЗДІЛ 3 Бджолиний алгоритм

Бджолиний алгоритм (алгоритм оптимізації наслідуванням бджолиної колонії, англ. artificial bee colony optimization, ABC) – один з поліноміальних евристичних алгоритмів для розв‘язку оптимізаційних задач в області інформатики та дослідження операцій. Відноситься до категорії стохастичних біонічних алгоритмів, заснований на імітації поведінки колонії медоносних бджіл при зборі нектару в природі. Ідея бджолиного алгоритму полягає в тому, що усі бджоли на кожному кроці будуть вибирати як елітні ділянки для дослідження, так і ділянки в околиці елітних, що дозволить, по-перше, урізноманітнити популяцію пошуків на наступних ітераціях, по-друге, збільшити ймовірність виявлення розв‘язку близького до оптимального.

3.1.Штучний бджолиний рій

Рій — це сукупність агентів або організмів; інтелект рою можна визначити як соціальну поведінку рою, в якій автономні індивіди взаємодіють один з одним у децентралізований та самоорганізований спосіб. Взаємодія особин покращує емпіричні знання про навколишнє середовище і приводить рій до оптимального стану.

Такий інтелект ми можемо спостерігати в природі. Наприклад, відомо, що мурахи знаходять найкоротший шлях від своєї колонії до джерела їжі. На початку люди досліджують різні напрямки від і до місця призначення. Коли знайдено сприятливий маршрут, мурахи позначають шлях феромонами – хімічними речовинами, які мурахи відкладають на землі. У міру того, як більше мурах йдуть тим же слідом, феромони посилюються, залучаючи більше мурах. Отже, більшість мурах йдуть і сходяться до найкоротшого шляху.

Штучний бджолиний рій (ABC) - один з найновіших алгоритмів, розроблений Дервісом Карабога у 2005 році, який надихнутий розумною поведінкою медоносних бджіл. Він такий же простий, як алгоритми оптимізації рою частинок (PSO) та диференційної еволюції (DE), і використовує лише загальні параметри керування, такі як розмір колонії та максимальна кількість циклів.

3.2. Алгоритм штучного бджолиного рою (ABC)

Алгоритм штучної бджолиної колонії (англ. Artificial Bee Colony algorithm, ABC) є метаевристичним алгоритмом, інспірованим поведінкою бджолиних роїв. Він був запропонований Карімом Абдель Разеком та спрямований на розв'язання оптимізаційних задач. Основна ідея полягає в тому, що агенти, що моделюють бджоли, працюють колективно, обмінюючись інформацією та розв'язуючи задачу оптимізації.

Карабога та Бастурк (2007) і Карабога, 2010, Карабога, 2005 розробили алгоритм ABC, моделюючи кормову поведінку медоносних бджіл. Наразі його використовували для вирішення багатьох проблем реального світу. Промислове машинобудування, машинобудування, електротехніка, електроніка, техніка керування, цивільне будівництво, розробка програмного забезпечення – це галузі, в яких він використовується.

3.3. Особливості алгоритму штучної бджолиної колонії

Особливості алгоритму штучної бджолиної колонії:

Популяція бджіл: У штучній бджолиній колонії існує популяція бджіл, які представляють потенційні розв'язки задачі. Вони можуть бути поділені на три види бджіл: розвідувачі, спостерігачі та джерела їжі.

Розвідувачі: Розвідувачі проводять дослідження нових розв'язків, випадково генеруючи нові значення параметрів задачі. Вони досліджують нові області простору розв'язків.

Спостерігачі: Спостерігачі стежать за якістю розв'язків, що виявляються розвідувачами. Якщо якість розв'язку є високою, вони взаємодіють з розвідувачами, щоб обмінятися інформацією.

Джерела їжі: Джерела їжі представляють розв'язки задачі, які вважаються потенційно хорошими. Якщо бджола знаходить кращий розв'язок, вона стає джерелом їжі і обмінюється цим розв'язком з іншими бджолами.

Функція оцінювання: Існує функція оцінювання, яка оцінює якість кожного розв'язку задачі. Вона використовується для порівняння та відбору кращих розв'язків.

3.4. Застосування ABC

Сфери застосування алгоритму штучної бджолиної колонії:

Оптимізація функцій: Алгоритм штучної бджолиної колонії може бути використаний для пошуку глобального мінімуму або максимуму функції.

Кластеризація: Використання алгоритму штучної бджолиної колонії дозволяє кластеризувати дані шляхом групування схожих об'єктів.

Нейромережеве навчання: Можна використовувати алгоритм для оптимізації ваг нейронної мережі шляхом підбору оптимальних значень параметрів.

Розкладання задач: Алгоритм може бути застосований для розкладання складних задач на менші підзадачі та оптимізації їх розв'язку.

Розміщення об'єктів: За допомогою алгоритму можна знайти оптимальне розташування об'єктів у просторі з урахуванням обмежень та критеріїв.

Це лише кілька прикладів сфер застосування алгоритму штучної бджолиної колонії. Завдяки своїй ефективності та здатності до знаходження глобальних оптимумів, цей алгоритм знайшов широке застосування в різних областях оптимізації та розв'язанні задач.

3.5. принцип роботи ABC

Алгоритм штучної бджолиної колонії (ABC), представлений Д. Карабогою, є одним із підходів, який використовувався для пошуку оптимального рішення в задачах чисельної оптимізації. Цей алгоритм заснований на поведінці медоносних бджіл у пошуках якісного джерела їжі.

Колонія досягає процвітання, за допомогою посилання робочих бджіл на поля з квітами. Є логічним, що, ділянки квітів зі значною кількістю нектару, який можна зібрати докладаючи менших зусиль, повинні бути відвіданими більшою кількістю бджіл, в той час як ділянки з відповідно меншою їх кількістю мають бути відвіданими меншою частиною бджіл.

Збір нектару розпочинається з вулика (колонії), коли бджоли розвідники відправляються на пошук так званих перспективних ділянок квітів. Ці бджоли рухаються довільним чином, переміщаючись з однієї ділянки квітів на іншу. За час збору нектару колонія продовжує проводити розвідку, завжди тримаючи певний відсоток популяції у ролі бджіл розвідників. При поверненні даних бджіл до вулика, ті бджоли-розвідники, які знайшли ділянку, що за своїми

якісними характеристиками оцінюється вище певного визначеного порогу (дана величина вимірюється як комбінація певних складових, серед яких основною є вміст цукру), залишають пробу нектару з цього джерела, і виходять на так званий «танцювальний майданчик» для виконання танцю, що є відомим як «танець з похитуванням».

Цей загадковий в біологічному сенсі танець має важливе значення для обміну інформацією в колонії, і містить у собі три порції інформації щодо квіткової ділянки: напрям, в якому розташована ділянка, відстань даної ділянки від вулика та міра її якості (або фітнес-функція). За допомогою цієї інформації популяція бджіл може відправити своїх робочих бджіл одразу ж до ділянок квітів, без використання для цього жодної іншої інформації.

Знання щодо навколишнього середовища кожною бджолою отримується виключно з даного танцю. Танець допомагає колонії виконати оцінку відносного внеску кожної з ділянок за якістю її нектару, та обсягом енергії, що необхідна для збирання нектару з неї. З завершенням виконання цього танцю, розвідник бере з собою бджіл-спостерігачів, що очікують для цієї задачі у вулику, і повертається до квіткової ділянки. Чим більш перспективною є ділянка, тим більше бджіл-спостерігачів буде відправлено до неї. Це дозволяє популяції збирати в більш ефективний та швидкий спосіб.

Колонія бджіл контролює рівень нектару в процесі його збирання з ділянки. Інформація про цей рівень буде використана в наступному танці при поверненні до вулика. У випадку високого показника якості ділянки, вона буде представлена під час наступного танцю й відповідно до неї буде залучено більшу кількість бджіл.

В даному алгоритмі бджолиної колонії власне колонія складається з 3 типів бджіл: робочих, спостерігачів та розвідників.

Спостерігачами називають тих бджіл, що очікують біля «майданчика для танців» наприйняття рішення щодо наступного джерела нектару для них.

Робочими бджолами називають тих бджіл, що виконують збір нектару з відвіданих ними раніше джерел.

Розвідниками називають тих бджіл, що виконують довільний пошук.

Алгоритм ABC поділяє колонію на дві частини, першою з яких є робочі бджоли, а другою є спостерігачі. Кожне джерело нектару обробляється лише однією робочою бджолою. Тобто кількість джерел в околиці вулика є рівною кількості робочих бджіл. Та робоча бджола, джерело якої більше не містить нектару, перетворюється на розвідника.

Загальна схема алгоритму ABC така:

```
Initialization Phase
REPEAT
  Employed Bees Phase
  Onlooker Bees Phase
  Scout Bees Phase
  Memorize the best solution achieved so far
UNTIL(Cycle=Maximum Cycle Number or a Maximum CPU time)
```

Тобто, алгоритм складається з наступних кроків:

1. Виконати ініціалізацію колонії.

2. Виконувати повторення:

a. Виконати розміщення робочих бджіл на джерелах нектару.

b. Виконати розміщення спостерігачів на тих самих джерелах нектару.

c. Для знаходження нових джерел нектару, відправити розвідників в область пошуку.

d. Повторювати кроки 1-2 допоки не виконані вимоги.

3.6. Етапи алгоритму штучного бджолиного рою

3.6.1. Етап 1

Кожна ітерація пошуку алгоритму ABC містить у собі 3 етапи: робочі бджоли відправляються до джерел нектару та виконують оцінку кількості нектару в даних джерелах; спостерігачі виконують вибір джерел нектару після обміну інформацією з робочими бджолами та визначення кількості нектару в джерелах; визначення, які з робочих бджіл перетворюються на розвідників, після чого виконується їх відправлення до потенційних джерел нектару. На стадії ініціалізації алгоритму бджоли довільним чином обирають джерело харчування та визначають кількість нектару у ньому.

Після цього бджоли повертаються до колонії і обмінюються інформацією щодо кількості нектару в квітах з бджолами-спостерігачами, що чекають на танцювальному майданчику.

3.6.2. Етап 2

На наступному етапі після обміну інформацією, всі робочі бджола переходять до області джерел нектару, які вони відвідувала на попередніх ітераціях, допоки ці джерела харчування існують в їх пам’яті, після чого бджоли вибирають нові джерела нектару поміж сусідніх.

3.6.3. Етап 3

На останньому етапі спостерігачі визначають найкраще з джерел харчування відповідно до інформації щодо нектару, інформація про який надається робочими бджолами. Що більшою є кількість нектару в джерелі, тим вищою є

ймовірність того, що дане джерело буде обране спостерігачем. Таким чином, танець робочих бджіл, що надають інформацію щодо більшої кількості нектару, приваблює більше спостерігачів до таких ділянок. По прибуттю до обраної ділянки, спостерігач в околиці пам'яті в залежності від візуальної інформації обирає нове джерело нектару. Візуальна інформація отримується відповідно до порівняння координат джерел нектару. Коли джерело нектару залишається бджолами, бджола-розвідник має визначити нове джерело довільним чином і бджоли мають переключитися на нього.

В даному алгоритмі координати кожного з джерел нектару є потенційними розв'язками задачі оптимізації, а кількість нектару в джерелах відповідає фітнес-функції (або якісній характеристиці) відповідного з розв'язків. Кількість потенційних координат є рівною кількості робочих бджіл та спостерігачів.

На першому етапі ABC алгоритм генерує початкову популяцію, що є довільно розподілена для SN розв'язків (координати джерел нектару). Кожен з розв'язків $xi(i = 1,2, \ldots, SN)$ являє собою D-вимірний вектором. Колонія після ініціалізації зазнає змін впродовж ітерацій $C = 1,2, \ldots, MCN$, під час процесів пошуку робочими бджолами, бджолами-спостерігачами та бджолами-розвідниками. Робочі бджоли змінюють свої координати в пам'яті залежно від локальної інформації (візуальної інформації) та досліджують кількість нектару нового джерела. У випадку, коли кількість нектару нового джерела є вищою за попередню, бджола запам'ятовує нові координати і видаляє з пам'яті попередні. В іншому випадку бджола залишає пам'ять в такому ж стані. Після завершення процесу пошуку усіма робочими бджолами, вони поширюють інформацію щодо нектару в джерелах та координати джерел зі спостерігачами на майданчику для танців.

Спостерігачі виконують оцінку інформації щодо нектару, яка отримана від усіх робочих бджіл та виконують вибір джерела нектару з ймовірністю, яка залежить від кількості нектару. Бджола-спостерігач змінює в своїй пам'яті

інформацію про джерело нектару та оцінює кількість нектару в потенційному джерелі таким же чином, як це виконують робочі бджоли. У випадку, коли нове джерело містить більше нектару за попереднє, бджола розташування цього нового джерела та видаляє з пам'яті інформацію про попереднє.

РОЗДІЛ 4 Основні особливості алгоритму штучного бджолиного рою

Особливості алгоритму штучного бджолиного рою:

Розвідники, диваки та прибиральниці: алгоритм передбачає поділ ролей між бджолами. Розвідники досліджують нові потенційні рішення, диваки аналізують інформацію, що обмінюється між бджолами, та вибирають кращі варіанти, а прибиральниці оновлюють рішення, що вже існують.

Множинність пошукових точок: кожна бджола зберігає свою власну пошукову точку або рішення, яке може бути розглянуте як потенційне оптимальне. Це дозволяє алгоритму досліджувати різні області простору рішень та уникати потрапляння в локальні мінімуми.

Взаємодія та обмін інформацією: бджоли обмінюються інформацією про якість своїх рішень та місцезнаходження. Це дозволяє швидко виявляти проміжні та потенційно кращі рішення, а також сприяє глобальній.

РОЗДІЛ 5 Реалізація ABC

```
import sys
import numpy as np
from deap.benchmarks import *
import progressbar

class ABC:

    def __init__(_self, conf):
        _self.conf = conf
        _self.foods = np.zeros((_self.conf.FOOD_NUMBER, _self.conf.DIMENSION))
        _self.f = np.ones((_self.conf.FOOD_NUMBER))
        _self.fitness = np.ones((_self.conf.FOOD_NUMBER)) * np.iinfo(int).max
        _self.trial = np.zeros((_self.conf.FOOD_NUMBER))
        _self.prob = [0 for x in range(_self.conf.FOOD_NUMBER)]
        _self.solution = np.zeros((_self.conf.DIMENSION))
        _self.globalParams = [0 for x in range(_self.conf.DIMENSION)]
        _self.globalTime = 0
        _self.evalCount = 0
        _self.cycle = 0
        _self.experimentID = 0
        _self.globalOpts = list()

        if (_self.conf.SHOW_PROGRESS):
            _self.progressbar = progressbar.ProgressBar(max_value=_self.conf.MAXIMUM_EVALUATION)
        if (not(conf.RANDOM_SEED)):
            random.seed(conf.SEED)

    def calculate_function(_self, sol):
        try:
            if (_self.conf.SHOW_PROGRESS):
                _self.progressbar.update(_self.evalCount)
            return _self.conf.OBJECTIVE_FUNCTION(sol)

    except ValueError as err:
        print(
            "An exception occured: Upper and Lower Bounds might be wrong. (" + str(err) + " in calculate_function)")
        sys.exit()

def calculate_fitness(_self, fun):
    _self.increase_eval()
    if fun >= 0:
        result = 1 / (fun + 1)
    else:
        result = 1 + abs(fun)
    return result

def increase_eval(_self):
    _self.evalCount += 1

def stopping_condition(_self):
    status = bool(_self.evalCount >= _self.conf.MAXIMUM_EVALUATION)
    if(_self.conf.SHOW_PROGRESS):
      if(status == True and not( _self.progressbar._finished )):
          _self.progressbar.finish()
    return status

def memorize_best_source(_self):
    for i in range(_self.conf.FOOD_NUMBER):
        if (_self.f[i] < _self.globalOpt and _self.conf.MINIMIZE == True) or (_self.f[i] >= _self.globalOpt and _self.conf.MINIMIZE == False):
            _self.globalOpt = np.copy(_self.f[i])
            _self.globalParams = np.copy(_self.foods[i][:])

def init(_self, index):
    if (not (_self.stopping_condition())):
        for i in range(_self.conf.DIMENSION):
            _self.foods[index][i] = random.random() * (_self.conf.UPPER_BOUND - _self.conf.LOWER_BOUND) + _self.conf.LOWER_BOUND
        _self.solution = np.copy(_self.foods[index][:])
        _self.f[index] = _self.calculate_function(_self.solution)[0]
        _self.fitness[index] = _self.calculate_fitness(_self.f[index])
```

```
Terminal:  Local  +
100% |##################################################################################################
2023-06-01160051-0 run:  644.5495550013936  Cycle:  1  Time:  3.3818359375
2023-06-01160051-1 run:  543.8901773662997  Cycle:  1  Time:  2.4541015625
2023-06-01160051-2 run:  441.70200798952555  Cycle:  1  Time:  3.49658203125
2023-06-01160051-3 run:  665.0914976633669  Cycle:  1  Time:  3.10986328125
2023-06-01160051-4 run:  666.4324481107623  Cycle:  1  Time:  3.218017578125
2023-06-01160051-5 run:  680.1424048548341  Cycle:  1  Time:  3.19287109375
2023-06-01160051-6 run:  571.4193776962118  Cycle:  1  Time:  3.021484375
2023-06-01160051-7 run:  566.8942766244776  Cycle:  1  Time:  2.994384765625
2023-06-01160051-8 run:  697.5289081137586  Cycle:  1  Time:  1.7724609375
2023-06-01160051-9 run:  565.2173991212068  Cycle:  1  Time:  8.601318359375
2023-06-01160051-10 run:  476.4568316979164  Cycle:  1  Time:  2.14111328125
2023-06-01160051-11 run:  568.8164416060446  Cycle:  1  Time:  1.8330078125
2023-06-01160051-12 run:  564.7867643723187  Cycle:  1  Time:  1.69580078125
2023-06-01160051-13 run:  641.0379118019863  Cycle:  1  Time:  1.73681640625
2023-06-01160051-14 run:  522.5823799143375  Cycle:  1  Time:  1.79443359375
2023-06-01160051-15 run:  458.62785354045417  Cycle:  1  Time:  1.73974609375
2023-06-01160051-16 run:  615.9002389037365  Cycle:  1  Time:  7.508056640625
2023-06-01160051-17 run:  518.6667254101317  Cycle:  1  Time:  1.68994140625
2023-06-01160051-18 run:  679.3605235822356  Cycle:  1  Time:  1.39404296875
2023-06-01160051-19 run:  720.1506990865777  Cycle:  1  Time:  0.9833984375
2023-06-01160051-20 run:  580.8301998852667  Cycle:  1  Time:  0.937744140625
2023-06-01160051-21 run:  605.8670496346551  Cycle:  1  Time:  0.974365234375
2023-06-01160051-22 run:  608.4767523862829  Cycle:  1  Time:  0.947021484375
2023-06-01160051-23 run:  703.6483643621827  Cycle:  1  Time:  0.944091796875
2023-06-01160051-24 run:  629.3645159497029  Cycle:  1  Time:  0.947998046875
2023-06-01160051-25 run:  644.3801837893853  Cycle:  1  Time:  0.955810546875
2023-06-01160051-26 run:  696.7630623470932  Cycle:  1  Time:  0.956787109375
2023-06-01160051-27 run:  652.5551235818742  Cycle:  1  Time:  6.219970703125
2023-06-01160051-28 run:  690.9585439267196  Cycle:  1  Time:  0.98193359375
2023-06-01160051-29 run:  580.0575583043528  Cycle:  1  Time:  0.957275390625
Mean:  606.7385258875032  Std:  73.96454679820191  Median:  612.1884956450097
```

ВИСНОВКИ

У даній роботі досліджується ройовий інтелект і алгоритм штучного бджолиного рою як метаевристичні підходи до розв'язання оптимізаційних задач. Ройовий інтелект базується на принципах колективного інтелекту, в якому група простих агентів спільно працює для досягнення глобальних цілей. Алгоритм штучного бджолиного рою, в свою чергу, моделює поведінку бджіл рою для ефективного пошуку оптимальних рішень. У роботі розглядаються особливості цих алгоритмів, їхні принципи роботи та сфери застосування.

Також в даній курсовій роботі було розглянуто візуалізацію алгоритму штучного бджолиного рою. Візуалізація алгоритму має важливі причини і застосування в сучасному контексті, включаючи розуміння та аналіз алгоритму, візуалізацію процесу оптимізації, навчання та освіту, а також подачу результатів та комунікацію. Ці аспекти підкреслюють важливість візуалізації для кращого розуміння, аналізу, навчання та комунікації в галузі алгоритмів штучного бджолиного рою. Візуалізація допомагає дослідникам і розробникам краще зрозуміти та аналізувати роботу алгоритму, відстежувати процес оптимізації, полегшує навчання новачків та сприяє ефективній подачі результатів та комунікації зі спільнотою. Таким чином, візуалізація алгоритму штучного бджолиного рою відіграє важливу роль у розвитку цієї галузі та сприяє його використанню у практичних застосуваннях.

СПИСОК ЛІТЕРАТУРИ

1. Карабога Д., Бастурк Б. (2007), Алгоритм оптимізації штучної бджолиної колонії (ABC) для розв'язання проблем оптимізації з обмеженнями, Досягнення м'яких обчислень: Основи нечіткої логіки та м'яких обчислень, том 4529/2007 LNCS: 789-798, Шпрінгер, Берлін.

2. Карабога, Д., Бастурк, Б., Озтурк, К. (2007), Алгоритм оптимізації штучної бджолиної колонії (ABC) для навчання прямої нейронної мережі, Моделювання рішень для штучного інтелекту, том 4617/2007 LNCS: 318-319 , Springer, Берлін.

3. Карабога Д., Акай Б. (2009), A Comparative Study of Artificial Bee Colony Algorithm, Applied Mathematics and Computation, 214, 108-132, Elsevier, Нідерланди.

4. Карабога, Д., Озтурк, К. (2009), Навчання нейронних мереж за допомогою алгоритму штучної колонії бджіл на основі класифікації шаблонів, Світ нейронних мереж, 19 (3), 279-292, Інститут комп'ютерних наук AS CR, vvi, Чеська Республіка.

5. Карабога, Д., Озтурк, К. (2010), Новий підхід до кластеризації: алгоритм штучної бджолиної колонії (ABC), Applied Soft Computing, Elsevier, Нідерланди, In Press.

6. Кеннеді, Дж., Еберхарт, Р.К. (1995), Оптимізація рою частинок, у матеріалах 1995 року Міжнародної конференції IEEE з нейронних мереж, 4: 1942–1948, Перт, Західна Австралія.

7. Афшар А. та ін. Алгоритм оптимізації спаровування медоносних бджіл (HBMO) для оптимальної роботи пласта. J. Franklin Inst. Б(2007)
8. Добре Б. Дослідження оптимізації роїв частинок і алгоритмів створення штучних бджолиних колоній для багаторівневого визначення порогів. апл. Soft Comput.(2013)
9. Akay B. та ін. Модифікований алгоритм штучної бджолиної колонії для оптимізації реальних параметрів. Інформ. наук.(2012)

10. Akay B. та ін. Опитування щодо варіантів алгоритму штучної бджолиної колонії для задач бінарного, цілочисельного та змішаного цілочисельного програмування. апл. Soft Comput.(2021)