https://doi.org/10.31891/2219-9365-2023-75-4
УДК 004.942:519.876.5

БАРАБАШ Олег
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
https://orcid.org/0000-0003-1715-0761
e-mail: bar64@ukr.net
КИР'ЯНОВ Артемій
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
https://orcid.org/0000-0003-3116-0122
e-mail: veniaminandater888@gmail.com

ДОСЛІДЖЕННЯ АЛГОРИТМІВ ПОВЕДІНКИ ЗГРАЙ У ПРИРОДІ ДЛЯ МОЖЛИВОСТІ ЗАСТОСУВАННЯ В ГРУПОВИХ ПОЛЬОТАХ БЕЗПІЛОТНИХ ЛІТАЛЬНИХ АПАРАТІВ

У статті було проведено докладний аналіз алгоритмів поведінки груп природних утворень, таких як зграї бджіл, риб і птахів, які спроможні діяти як єдине ціле в природному середовищі. Досліджено здатність окремих особин успішно координувати свої дії в групах. Однією з ключових особливостей цих природних систем є їх здатність до координації і співпраці. Зграї бджіл, косяки риб та зграї птахів можуть виконувати складні колективні дії, такі як пересування, полювання, збирання їжі, та вирішувати різноманітні завдання як єдина інтегрована система. На основі аналізу природних механізмів, що лежать в основі алгоритмів взаємодії, розглянуто можливість застосування у різних галузях, таких як галузь інформатики та робототехніки. В інформатиці, використання аналогії з поведінкою зграї може мати важливе значення для оптимізації комп'ютерних мереж, розробки алгоритмів маршрутизації даних, а також для вирішення завдань розподілених обчислень. У робототехніці ці алгоритми можуть бути використані для створення безпілотних літальних апаратів (БПЛА), які здатні працювати як частинки зграї. Застосування зазначених алгоритмів дозволить розробити БПЛА, які можуть ефективно координувати свої дії, виконувати пошук та рятування, моніторинг або інші завдання у груповому режимі. Це може мати суттєве значення для застосування в сферах, таких як цивільна авіація, безпека, моніторинг навколишнього середовища та багато інших. Досліджено застосування цих алгоритмів в управлінні елементів транспортних системам. Великі зграї бджіл, косяки риб та зграї птахів демонструють здатність до ефективного руху та координації в умовах обмеженого простору та ресурсів. Проведено порівняльний аналіз алгоритмів взаємодії частинок зграй та надано практичні рекомендації щодо застосування того чи іншого алгоритму в різних умовах групового польоту БПЛА. Отримані результати досліджень можна застосувати при розробці систем управління окремими безпілотними літальними апаратами, які виконують завдання особисто та в групі.

Ключові слова: зграї в природі, безпілотний літальний апарат, поведінка колонії бджіл, алгоритм зграї птахів, агенти, бази даних, штучний інтелект, архітектура програмного забезпечення.

BARABASH Oleg, KYRIANOV Artemii
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

RESEARCH OF ALGORITHMS OF FLOCK BEHAVIOR IN NATURE FOR THE POSSIBILITY OF APPLICATION IN GROUP FLIGHTS OF UNMANNED AIRCRAFT

The article carried out a detailed analysis of the algorithms of the behavior of groups of natural formations, such as swarms of bees, fish and birds, which are able to act as a single unit in the natural environment. The ability of individuals to successfully coordinate their actions in groups was studied. One of the key features of these natural systems is their ability to coordinate and cooperate. Swarms of bees, shoals of fish, and flocks of birds can perform complex collective actions such as locomotion, hunting, gathering food, and solve a variety of tasks as a single, integrated system. Based on the analysis of the natural mechanisms underlying interaction algorithms, the possibility of application in various fields, such as the field of informatics and robotics, is considered. In computer science, using an analogy with swarm behavior can be important for optimizing computer networks, developing data routing algorithms, and solving distributed computing problems. In robotics, these algorithms can be used to create unmanned aerial vehicles (UAVs) that are capable of operating as swarm particles. The application of these algorithms will allow the development of UAVs that can effectively coordinate their actions, perform search and rescue, monitoring or other tasks in a group mode. This can be essential for applications in areas such as civil aviation, security, environmental monitoring and many others. The application of these algorithms in the management of elements of transport systems has been studied. Large swarms of bees, shoals of fish, and flocks of birds demonstrate the ability to move efficiently and coordinate under conditions of limited space and resources. A comparative analysis of swarm particle interaction algorithms was carried out and practical recommendations were given for the application of one or another algorithm in different conditions of UAV group flight. The obtained research results can be applied in the development of control systems for individual unmanned aerial vehicles that perform tasks individually and in a group.

Keywords: flocks in nature, unmanned aerial vehicle, bee colony behavior, bird flock algorithm, agents, databases, artificial intelligence, software architecture.

Постановка проблеми у загальному вигляді та її зв'язок із важливими науковими чи практичними завданнями

Вивчення алгоритмів поведінки зграї в природі стає дедалі актуальнішим у контексті розвитку сучасних технологій, зокрема безпілотних літальних апаратів (БПЛА), їх застосування для цивільних та

військових завдань. Важливими науковими та практичними завданнями є одночасне застосування кількох БПЛА, або певної сукупності БПЛА, з єдиним задумом в просторі та в часі. Така сукупність безпілотних літальних апаратів в сучасній технічній літературі отримала назву групи БПЛА або, за аналогією із біологічними істотами, – зграї БПЛА. Об'єднання в зграї БПЛА дозволяє керувати кожним окремим апаратом за допомогою автоматизованої системи, або використовувати програми з елементами штучного інтелекту, або з кількома операторами.

На сьогоднішньому етапі розвитку безпілотної авіації актуальними є дослідження алгоритмів поведінки зграї в природі та їх застосування для безпілотних літальних апаратів, аналіз ключових аспектів зазначених питань, а також можливість подальшого використання отриманих результатів досліджень. Цим питанням і присвячена дана стаття.

Аналіз досліджень та публікацій

Термін «Ройовий інтелект» був введений вченими Ван Цзінь і Херардо Бені в 1989 році. Поняття ройового інтелекту все тісніше переплітається з алгоритмами обробки і оптимізації великих кількостей даних і потоків інформації [1-2].

Люди стали цікавитися «ройовою поведінкою» ще давно. Вони вивчали яким чином птахи летять на південь величезними косяками, не збиваючись з курсу; як бджоли можуть так точно визначати і добувати для всієї колонії харчування.

Всі ці великі групи тварин (комах) можна об'єднати одним загальним словом – рій [3-4]. Спостерігаючи за різними природними прикладами роїв, людство придумало різні моделі ройового інтелекту. Їх поведінка ґрунтувалася на різних шляхах взаємодії з навколишнім середовищем і між собою [4].

Модель ройового інтелекту має на увазі наявність так званої «мультиагентної системи». Вона визначається як система, що складається з множини інтелектуальних агентів – програм, здатних самостійно, протягом деякого, досить тривалого проміжку часу, виконувати поставлене завдання [5-6].

Алгоритм бджолиного рою – алгоритм ройового інтелекту, заснований на імітації поведінки колонії бджіл, може використовуватися в задачах оптимізації. Необхідною умовою для його застосування є наявність деякої топологічної відстані або аналогічної величини в області рішень [6-7].

Алгоритм FSS заснований на харчовій поведінці косяку риб, які пересуваються з метою пошуку їжі, що є оптимальним рішенням, в межах області пошуку. При харчуванні вага кожної риби збільшується та формалізує її індивідуальний успіх в пошуку рішення та відіграє роль пам'яті риби. При знаходженні ефективного місця харчування риби об'єднуються у косяки [8-10].

Таким чином, актуальність зазначених досліджень обгрунтовується тим, що на сьогоднішній день ефективним напрямком в еволюційному моделюванні є імовірнісні алгоритми, засновані на процесах, що відбуваються в живій природі. Проектуючи закономірності навколишнього світу на певні сфери діяльності людини, такі як соціальні, технічні, політичні, ми отримуємо ефективний інструмент для вирішення завдань в різних напрямках діяльності людини. Особливо важливо вивчати поведінку окремих особин в зграї та їх скоординований рух для можливого застосування в системах управління груповими польотами БПЛА.

Формулювання цілей статті

Мета дослідження. Метою даної статті є виявлення та використання природних алгоритмів для керування безпілотними літальними апаратами з метою покращення їх ефективності та безпеки за різних сценаріїв застосування.

Виклад основного матеріалу

1. АЛГОРИТМ ПОВЕДІНКИ ЗГРАЇ БДЖІЛ

1.1. Загальні відомості

Математична модель алгоритму, заснованого на поведінці колонії бджіл. Поведінка комах в живій природі полягає в тому, що спочатку з вулика вилітає у випадковому напрямку певна кількість бджіл-розвідників, які намагаються відшукати ділянки, де є нектар (рис. 1). Через якийсь час бджоли повертаються у вулик і особливим чином повідомляють іншим, де і скільки вони знайшли нектару. Після цього на знайдені ділянки відправляються інші бджоли, причому, чим більше на даній ділянці передбачається знайти нектару, тим більше бджіл летить в цьому напрямку [1].

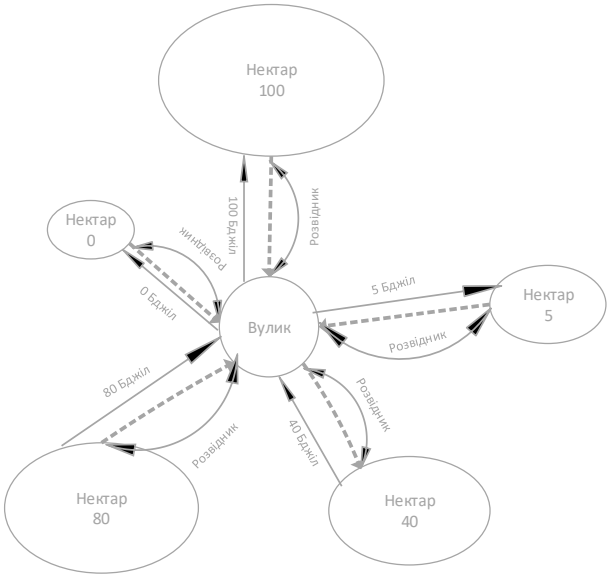

Рис. 1. Схема поведінки колонії бджіл в живій природі

1.2. Сутність алгоритму

На рис. 2 приведена модель, заснована на поведінці колонії бджіл. Тут В є вершиною зоряного графа. Решта вершини - розвідані області зі значеннями цільової функції (ЦФ): вершина А – ЦФ 100, вершина F – ЦФ 80, вершина Е – ЦФ 40, вершина С – ЦФ 5, вершина D – ЦФ 0. Інформацію про значення цільової функції в цих вершинах бджоли розвідники (агенти-розвідники) передають робочим бджолам (агентам). Після цього з вершини В прямує кілька агентів в інші вершини для дослідження їх околиць. Кількість агентів в кожному напрямку пропорційно значенню цільової функції кожної вершини графа.

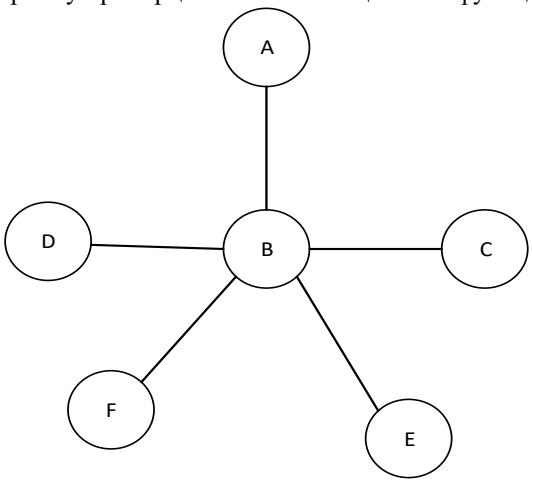

Рис. 2. Модель, заснована на поведінці колонії бджіл

Що стосується завдання оптимізації (завданням знаходження мінімуму або максимуму) формалізуємо деякі поняття. Варто зазначити, що пошук здійснюється не на нескінченності, а на заздалегідь заданому відрізку [a, b]. Позначимо N – число ділянок (блоків). Для спрощення візьмемо кількість блоків N = const, проте кількість ділянок може змінюватися динамічно в процесі роботи алгоритму. Уявімо розташування ділянок поля у вигляді множини X = {x1, x2, ..., xi, ... xN}, а значення цільової функції на кожній ділянці у вигляді множини F (X) = {f (x1), f (x2), ..., f (xi), ..., f (xN)}. Зафіксуємо загальну кількість бджіл B = const. Це дозволить скоротити нам обсяг обчислювальних ресурсів. Таким чином, завжди є можливість точно призначити певну кількість бджіл в певний блок, пропорційно значенню цільової функції [2].

$$b_i = B * (1 - \frac{f(x_i) - \min(F(X))}{\max(F((X)) - \min(F(X))}) \quad (1)$$

Для завдання максимізації:

$$b_i = B * (\frac{f(x_i) - \min(F(X))}{\max(F(X)) - \min(F(X))}) \quad (2)$$

де max (F (X)) – максимальне значення з множини F (X), min (F (X)) – мінімальне значення з множини F (X). Для формування пошуку в ширину в кожному блоці необхідно визначити околицю цього блоку. Визначимо околицю точки xi наступним чином:

$$[x_i - p_i, x_i + q_i] \tag{3}$$

де p_i – відхилення вліво, q_i- відхилення вправо. Параметри околиці розраховуються наступним чином:

$$p_i = b_i * \frac{|x_i - x_i| - 1|}{(\frac{(b_i + b_{i-1})}{2})} \tag{4}$$

$$q_i = b_i * \frac{|x_i - x_{i+1}|}{(\frac{(b_i + b_{i+1})}{2})} \tag{5}$$

Так як N = const, то не доцільно розширювати відрізок пошуку [a, b]. Тому введемо обмеження на p і q. p0 = 0, qN = 0.

1.3. Опис алгоритму

Наведемо алгоритм знаходження оптимальної цільової функції:

1. Задаємо відрізок [a, b]. Вибираємо на вихідному відрізку N значення аргументу і вважаємо значення функції f для кожного із значень множини X. Таким чином формуємо множини X та F (X).
2. Призначаємо бджіл в околиці точок x_i пропорційно значенням функції f (x) використовуючи формулу (1) або (2) відповідно до типу екстремуму. Для більш перспективних точок кількість бджіл в околиці виявиться більше, ніж у менш перспективних.
3. Розбиваємо пошуковий простір на області (рис. 3).
4. Розраховуємо величини околиці i-ої точки за формулами (3), (4) і (5).
5. У кожної околиці точки x_i, що належить інтервалу (xi-pi, xi + qi) вибору bi довільних точок з дискретністю не більше, ніж

$$\frac{(p_i + q_i)}{B} \tag{6}$$

Виходить множина Ti = {x'1, ..., x'j, ... x'bi}, де x'j ε [xi-pi, xi + qi].

6. Якщо для множини Ti існує, такий член x'j, то в множині X проводимо заміну.

$$f(x_j^{`}) < f(x_i) x_i = x_j^{`}$$

7. Якщо умови зупинки алгоритму не виконані, то переходимо до пункту 2. Інакше до пункту 8. Умовами зупинки можуть бути:
   - ♦ досягнення заданого числа ітерацій;
   - ♦ вироблення встановленого часу роботи алгоритму;
   - ♦ досягнення прийнятного значення цільової функції.
8. Кінець роботи алгоритму [3]

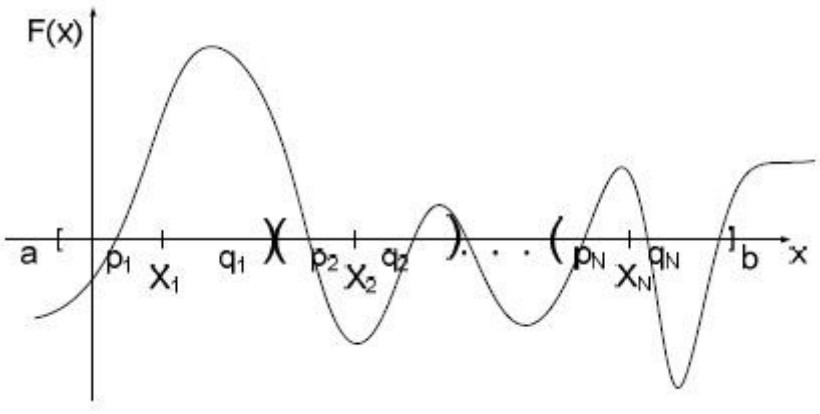

Рис. 3. Розбиття пошукового простору на області [3].

Слід відзначити, що особливістю розробленого алгоритму є здатність динамічно розбивати пошуковий простір на області, що зменшує час роботи алгоритму. Даний алгоритм ілюструє стратегію пошуку «Розділяй і володарюй», тобто проводиться розбиття складних оптимізаційних завдань на підзадачі. Головною перевагою є той факт, що, завдяки пошуку по всій довжині відрізка, значно знижується ймовірність попадання в локальний оптимум, а за рахунок розпаралелювання зменшується час. На кожній ітерації він дорівнює часу пошуку в певному блоці.

1.4. Переваги та недоліки

Переваги

Даний алгоритм дозволяє розділити процес розміщення елементів, ефективно управляти пошуком, отримувати оптимальні та квазіоптимальні рішення. Проведено обчислювальний експеримент. Результати проведених тестів та експериментів дозволили уточнити теоретичні оцінки часової складності алгоритмів проектування та їх поведінку для схем різної структури. У кращому випадку тимчасова складність алгоритму $\approx O(n \cdot \log(n))$, в гіршому випадку – $O(n^3)$.

Недоліки

Недолік даного методу – велике число вільних параметрів, від значення якого, як правило, залежить результат, з іншого боку, відсутні підстави для використання цих значень параметрів.

Висновок

Таким чином, алгоритм використовує стратегію "Розділяй і володарюй" для пошуку оптимальних рішень, уникнення локальних мінімумів та зменшення часу завдяки розпаралелюванню. Але він має недоліки через велику кількість параметрів без чітких підстав для їх вибору.

2. АЛГОРИТМ ПОВЕДІНКИ КОСЯКІВ РИБ

2.1. Загальні відомості

З середини минулого століття проводяться дослідження по симуляції біологічних механізмів природи, зокрема, пов'язані з процесом еволюції. Тільки в 80-х роках почалися практичні випробування цих методів в зв'язку з виниклою необхідністю в ефективному способу оптимізації n-арних функцій, що мають високу обчислювальну складність, багато екстремумів тощо. Говорячи про термінологію, варто згадати, що дані алгоритми відносяться до класу стохастичних пошукових. У багатьох джерелах також можна зустріти такі визначення, як поведінковий, інтелектуальний, метаевристичний або популяційний [4].

2.2. Сутність алгоритму

Даний алгоритм запропонували в 2008 році Філо (B. Filho) і Нето (L. Neto). Як і у всіх популяційних алгоритмах, в якості вхідних параметрів задаються: функція пристосованості (функція, для якої необхідно знайти екстремуми), область дослідження цієї функції і параметри роботи алгоритму, які будуть описані пізніше. У поточному алгоритмі FSS (Fish School Search) область пошуку являє собою акваріум, в якому плавають риби (агенти). Як відомо, в процесі пошуку їжі риби плавають косяком, тому в даному випадку кінцевою метою є усунення всіх агентів в області екстремуму функції. У загальних рисах схема роботи алгоритму наступна:

1. Ініціалізація популяції (рівномірний розподіл риб в акваріумі).
2. Міграція агентів до джерела їжі (аналогія: чим більший крок агенти зробили в напрямку області екстремуму функції, тим більше їжі вони отримали).
3. Завершення пошуку.

2.3. Опис алгоритму

Стадія «Міграція агентів» виконується ітераційно, і в кожній з ітерацій присутні оператори двох груп:

1. Оператори плавання, що забезпечують міграцію агентів в межах акваріума.
2. Оператори годування, здатні фіксувати успіх дослідження тих чи інших областей акваріума.

Процес виконання алгоритму, як уже було сказано вище, залежить від вхідних параметрів, що задаються користувачем. Вони характерні для всього акваріума в цілому.

populationSize – розмір популяції (кількість риб в косяку);

iterationCount – кількість ітерацій в стадії «Міграція агентів»;

lowerBoundPoint – верхня межа пошуку;

higherBoundPoint – нижня межа пошуку;

individStepStart – задає початковий радіус пошуку їжі навколо агентів;

individStepFinal – задає кінцевий радіус пошуку їжі навколо агентів;

weightScale – максимальна вага агента.

З їх допомогою регулюється співвідношення «точність – час роботи» алгоритму[5]. Що стосується самих агентів, то вони характеризуються двома величинами:

swimStatePos – позиція агента в різних стадіях плавання;

weight – поточний вага агента.

Нижче наведено псевдокод даного алгоритму:

```
initialize_randomly_all_fish;
while (stop_criterion is not met)
{
  for (each_fish)
  {
  individual_movement;
  evaluate_fitness_function;
  }
 feeding_operator;
  for (each_fish)
    instinctive_movement;
 calculate_barycentre;
  for (each_fish) do
  {
     volitive_movement;
     evaluate_fitness_function;
  }
  update_individidual_step;
}
```

Зауваження: всі нижченаведені пояснення роботи алгоритму розраховані на те, що вирішується завдання умовної максимізації функції. Разом із тим, це не повинно викликати сумнівів з приводу працездатності даного методу під час пошуку мінімальних значень функції.

1. initialize_randomly_all_fish

В даному алгоритмі риби плавають в акваріумі, розмірність якого дорівнює арності функції, і має межі, що збігаються з областю пошуку максимуму функції. Після введення параметрів алгоритму відбувається заповнення акваріума рибами, тобто кожному примірнику присвоюються випадкові координати (swimStatePos [0]), рівномірно розподілені в межах кордонів акваріума, і встановлюється вага, що дорівнює половині від максимального (weight = weightScale / 2) [6].

2. stop_criterion

У загальному випадку критерієм зупинки може бути досягнення певної точності результатів, час виконання програми, кількість використовуваної пам'яті та інше. У поточній реалізації алгоритму цим є кількість ітерацій, яка задається користувачем до початку виконання алгоритму (iterationCount).

3. individual_movement

У цій стадії плавання рибам дається шанс зробити індивідуальні переміщення, які обмежуються параметром «крок індивідуального плавання» (individStep): swimStatePos [1] = swimStatePos [0] + rand (-1; 1)*indivitStep

Якщо в результаті це переміщення виводить агента (рибу) за межі області пошуку або призводить до гіршого значенням функції, то вважається, що його становище не змінилося.

4. feeding_operator

Тепер необхідно закріпити успіх в індивідуальній стадії плавання. Для цього використовується характеристика «вага». Вона дорівнює зміні функції пристосованості для даного агента до і після індивідуальної стадії, нормованої максимальною зміною функції серед популяції:

weight + = Δfi / maxΔf

Взагалі кажучи, це є відмінною рисою даного алгоритму, так як не потрібно запам'ятовувати кращих агентів на попередніх ітераціях. Значення weight лежить в діапазоні [1; weightScale] і при необхідності коригується.

5. instinctive_movement

Після цього риби здійснюють наступну стадію плавання - інстинктивно-колективну. Для всього косяка риб вираховується величина «спільний крок міграції»:

$$m = \frac{\sum_{i=1}^{populationSize}((swimStatePos_i[1] - swimStatePos_i[0]) * \Delta f_i}{\sum_{i=1}^{populationSize} \Delta f_i} \quad (7)$$

З практичної точки зору це означає, що на кожного агента впливає вся популяція в цілому, при цьому вплив окремого агента пропорційно його успіхам в індивідуальній стадії плавання. Після цього вся популяція зміщується на нараховану величину m:

swimStatePos [2] = swimStatePos [1] + m

6. calculate_barycentre

Перед наступною операцією плавання необхідно виконати проміжні дії, а саме: підрахувати центр тяжіння всього косяка:

$$m = \frac{\sum_{i=1}^{populationSize}(swimStatePos_i[2] - weight_i)}{\sum_{i=1}^{populationSize} weight_i} \quad (8)$$

7. volitive_movement

На цьому етапі виконується колективно-вольова стадія плавання (рис. 4). Перш за все необхідно дізнатися, як змінилась вага популяції в порівнянні з попередньою ітерацією. Якщо вона збільшилась, значить популяція наблизилася до області максимуму функції, тому необхідно звузити коло його пошуку – тим самим виявляються особливі властивості. І навпаки: якщо вага косяка зменшилась, значить агенти шукають максимум не в тому місці, тому необхідно змінити напрямок траєкторії на протилежний і проявити диверсифікаційні властивості [7].

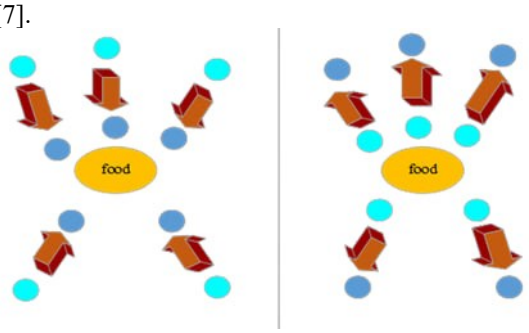

Рис. 4. Колективно-вольова стадія плавання

Величина collStep в такій формулі відповідає за крок зміщення у колективно-вольовій стадії. Рекомендується використовувати значення, в 2 рази більше індивідуального кроку пошуку. Оператор dist вираховує відстань між двома точками в евклідовому просторі:

$$\text{swimStatePos}[3] = \text{swimStatePos}[2] \pm \text{collStep} * \text{rand}(0;1) * * \frac{\text{swimStatePos}[2] - \text{barycentre}}{\text{dist}(\text{swimStatePos}[2];\ \text{barycentre})} \quad (9)$$

Зауваження: змінні, що відповідають за стан агентів, розглядаються як математичні n-мірні точки. Однак для здійснення над ними таких операцій, як складання / віднімання двох точок, додавання / віднімання точки і числа, множення / ділення точки і числа, а також порівняння точок, зручно розглядати змінні як радіус-вектори з початком координат в точці (0; 0) [8].

9. update_individidual_step

Останнім оператором в ітерації є лінійне зменшення кроку індивідуального пошуку для наступної ітерації. Ця дія є вже модифікацією стандартного алгоритму FSS для підвищення ефективності пошуку і виконується за такою формулою:

$$\text{individStep} -= \frac{\text{individStepStart} - \text{individStepFinal}}{\text{iterationCount}} \quad (10)$$

З практичної точки зору це пояснюється тим, що чим більше ітерацій вже пройдено, тим ближче агенти знаходяться до максимуму функції, і, отже, має сенс обмежити область пошуку на наступній ітерації [9].

2.4. Переваги та недоліки

Переваги

Кожна частинка виконує кілька видів переміщень:

- переміщення на підставі індивідуального досвіду;
- переміщення на підставі досвіду всього рою.

Недоліки

Переміщення на підставі досвіду рою складаються з двох фаз. На відміну від алгоритму рою частинок, при переміщеннях безпосередньо враховуються результати лише минулої ітерації. Важливо, що для даного алгоритму критерій повинен бути невідємним на всьому просторі пошуку: $f(X) \geq 0$, $X \in D$

Висновок

Таким чином, симуляція поведінки косяка риб, відома як алгоритм Fish School Search (FSS), була розроблена на основі досліджень із симуляцією біологічних механізмів природи та процесу еволюції. Подібно до реальних риб, агенти в цьому алгоритмі рухаються косяком в пошуках їжі або мінімуму функції. Алгоритм має свої переваги, такі як здатність кожної риби виконувати різні види переміщень, а також можливість збільшення швидкості пошуку більш високої якості рішень зі збільшенням кількості ітерацій.

3. ДОСЛІДЖЕННЯ АЛГОРИТМУ ПОЛЬОТУ ЗГРАЙ ПТАХІВ

3.1. Загальні відомості

Мурмурація – явище скоординованого польоту великої зграї птахів (шпаків, галок, ворон і т. д.), що утворюють динамічні об'ємні фігури змінної щільності.

Зрозуміти, яким чином алгоритм поведінки окремого птаха призводить до подібного явища, допомагає комп'ютерне моделювання. Так, воно показало, що механізм мурмурації грає роль захисту зграї від хижаків.

Аналогічне явище демонструють великі косяки риб. В загальному розумінні, мурмурація – це скоординований рух великих зграй птахів, які створюють динамічні об'ємні фігури зі змінною щільністю. Слід зазначити, що цей феномен залишається загадковим, і у вчених немає загальної згоди щодо його сутності, особливо щодо його мети.

3.2. Сутність алгоритму

По-перше, коли розглядають питання щодо переміщення зграї птахів, можна згадати перелітних птахів, які переважно рухаються у формації клину. Такий спосіб руху передбачає використання вихорів, створених передніми, сильнішими птахами, і це дозволяє ефективно зберігати енергію.

Важливою особливістю мурмурації є те, що в будь-який момент часу кожен шпак "бачить" (або відчуває) всіх інших шпаків у зграї, і ця кількість може сягати тисяч особин. Кожен птах при прийнятті своїх наступних рішень керується положеннями всіх членів зграї, а не лише найближчими сусідами. Проте питання чи слід враховувати положення всіх птахів, чи лише найближчих сусідів, залишається предметом дискусій. У випадку керування групою БПЛА, оптичне розпізнавання можливе тільки для найближчих сусідів, оскільки жодні маркери або інші оптичні ідентифікатори не можуть надійно розрізнятися через схожих об’єктів, розташованих набагато ближче до БПЛА, який виконує розпізнавання.

3.3. Опис алгоритму

Під час руху малого ґвинтокрилого БПЛА аеродинамічна ситуація відрізняється від руху птаха через меншу турбулентність, оскільки у першому випадку число Рейнольдса (яке може бути виражене, наприклад, через кінематичну в'язкість) приблизно складає:

$$Re_{\text{БПЛА}} = \frac{ul}{\nu} = \frac{25*0{,}1}{1{,}5*10^{-5}} = 1{,}6 * 10^5, \quad (11)$$

де u = 25 м/с – досить велика, як для малого БПЛА швидкість, взята з метою виконання більш обережних порівняльних оцінок;

l = 0,1 м – характерні розміри малого БПЛА, прийняті раніше;

$\nu = 1{,}5 * 10^{-5}$ м$^2/с$ – кінематична в’язкість повітря при нормальних умовах.

Число Рейнольдса для звичайного журавля буде приблизно таким:

$$Re_{\text{жур}} = \frac{ul}{\nu} = \frac{14*1}{1{,}5*10^{-5}} = 9{,}3 * 10^5, \quad (12)$$

де u = 50 км/год ≈ 14 м/с – середня швидкість польоту журавля;

l = 1 м – характерні розміри птаха.

Отже, отримане значення для птаха близько в 6 разів перевищує значення для БПЛА. Це свідчить про те, що для БПЛА використання вихорів є значно менш ефективним, навіть якщо припустити ідентичну аеродинаміку їхнього польоту (незважаючи на очевидні відмінності в аеродинамічних процесах між ґвинтокрилими літальними апаратами та птахами з крилами, які працюють на зони підвищеного тиску). Таким чином, стрій "клином" не є дуже ефективним для БПЛА.

Зовсім іншу картину ми спостерігаємо, розглядаючи захопливе явище, яке реалізують великі зграї деяких видів птахів, таких як шпаки, галки та ворони (рис. 5).

Рис. 5. Зразки мурмурації, що часто спостерігають у зграях шпаків [8].

Деякі наукові дослідження показують, що мурмурація допомагає зграям ефективніше ухилятися від хижаків. Такий рух змушує хижаків розсіювати свою увагу через велику кількість швидко рухаючихся об'єктів, і внаслідок цього стає майже неможливо влучити в окрему пташку в такій динамічній зграї. Звідси випливає логіка, що при регулярних і передбачуваних рухах зграєю, хижакам було б набагато легше адаптуватися і вивчити траєкторії окремих пташок, що спрощує їх захоплення.

Такі ж думки можна застосовувати і до перехоплювачів автоматичної дії із самонаведенням. У таких ситуаціях існує завжди конкретний метод для попадання ракети або керованого снаряду у рухому ціль або об'єкт. Для прикладу: виявлення теплового сліду, виявлення максимуму електромагнітних випромінювань або акустичних сигналів тощо. Проте, у випадку, коли існує багато БПЛА, і вони безперервно рухаються, вибір кінцевої цілі постійно змінюється, що може призвести до випадкового спрацьовування перехоплювачів, їхньої відмови або, ймовірніше, зіткнення з випадковими перешкодами у навколишньому середовищі [10].

Отже, корисним аспектом, який слід враховувати при розробці поведінки групи БПЛА, є їх взаємодія з перехоплювачами. У інших аспектах явище мурмурації залишається малодослідженим для активного використання в біонічних системах.

3.4. Переваги та недоліки

Переваги

Один з ефективних поліноміальних алгоритмів – алгоритм для знаходження наближених рішень задачі комівояжера, а також вирішення аналогічних завдань пошуку маршрутів на графах. Суть підходу полягає в аналізі та використанні моделі поведінки птахів, що шукають шляхи. Цей підхід являє собою метаевристічну оптимізацію. Алгоритми задачі комівояжера (ЗК) важко реалізувати за умови значної кількості вузлів мережі та внаслідок експоненціального часу конвергенції.

Він працює краще, ніж інші оптимізаційні алгоритми пошуку глобальних мінімумів для ЗК (нейронна мережа, генетичний алгоритм, тощо).

Особливості у порівнянні з генетичним алгоритмом (GA – Genetic Algorithms):

- залежить від пам'яті всієї зграї, а не лише пам'яті попередників;
- менш схильний до прийняття ранніх рішень про смугу (через випадковий вибір шляху).

Може використовуватися в динамічних додатках (наприклад, адаптація до змін відстані). Застосовується до різних завдань.

Недоліки

- Складний теоретичний аналіз, випадкові (не незалежні) рішення у результаті.
- Потенційний перерозподіл змін.
- Дослідження більш досвідчене, ніж теорія.
- Гарантоване підключення, але без вказаного терміну.
- Загалом, потрібно використовувати додаткові методи, такі як внутрішній пошук.
- Багато в чому залежить від визначених параметрів, вибраних лише експериментом [11].

Висновок

Таким чином, алгоритм поведінки зграї птахів може бути важливим для розвитку систем автоматичної навігації та координації груп безпілотних апаратів. Використання евристичних підходів, які

імітують природну поведінку птахів, може допомогти оптимізувати рішення в умовах змінного середовища. Однак, для успішної реалізації цих підходів, необхідно вдосконалити параметри та адаптувати їх до конкретних умов та завдань.

4. ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА ОСОБЛИВОСТЕЙ ЗАСТОСУВАННЯ АЛГОРИТМІВ

Зіставлення описів даних алгоритмів ройового інтелекту дозволяє виділити принципи, які використовує кожен з алгоритмів, крім загальних принципів ройового інтелекту.

Таблиця 1

Індивідуальні особливості ройових алгоритмів

Алгоритм	Особливості
Рій бджіл	- частина агентів виконує повністю випадковий пошук, не користуючись своїм та колективним досвідом; - заборона на збіжність всіх агентів в одному простору пошуку рішень.
Косяк риб	- декілька варіантів на етапі переміщення частинок; - використання градієнта; - використання «центру ваги» частинок.
Зграя птахів	- позитивний зворотній зв'язок; - рішення у вигляді шляху на графі; -зниження важливості досвіду, отриманого на минулих ітераціях згодом

Висновки з даного дослідження і перспективи подальших розвідок у даному напрямі

Завдяки зазначеним особливостям, для вирішення певних класів задач можна створювати ефективні гібридні алгоритми ройового інтелекту. Для завдань оптимізації на графах правильним буде вибір алгоритму зграї птахів, оскільки він по своїй природі полягає в пошуку певного маршруту на графі. Для завдань безперервної нединамічної оптимізації, в якій потрібно знайти найкраще рішення та мати достатній запас часу, підійдуть алгоритми, які використовують градієнтний пошук: косяка риб, оскільки їхні частки сходяться в екстремумі, їхнє переміщення зменшується, алгоритм починає виконувати локальний пошук та прямує до екстремуму. У завданнях динамічної оптимізації, навпаки, доцільно використовувати алгоритми рою бджіл і зграї птахів, оскільки в них передбачені механізми захисту від розбіжності всіх частинок: заборона на переміщення всіх частинок однієї ітерації та випадковий пошук (алгоритм рою бджіл), зниження вагових коефіцієнтів знайдених раніше шляхів за рахунок динамічного зменшення коефіцієнта посилення зворотного зв'язку (алгоритм зграї птахів). Завдяки такому вибору алгоритми здатні швидше знаходити нові рішення при зміні умов завдання.

References

1. Austin, R. Unmanned aircraft systems: UAVS design, development and deployment, 2011. P. 249 – 257.
2. Hildmann, Hanno, et al. Termite algorithms to control collaborative swarms of satellites. In: Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation, 2018. P. 109 – 124.
3. Brooks, Rodney. A robust layered control system for a mobile robot. IEEE journal on robotics and automation, 1986. P. 14 – 23.
4. Montgomery, James F.; Bekey. Learning helicopter control through "teaching by showing". In: Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No. 98CH36171). IEEE, 1998. P. 3647 – 3652.
5. Marsh, L., et al. Multi-agent UAV path planning. Proceedings. Modelling and Simulation Society of Australia & New Zealand, Melbourne, 2005. P. 49 – 67.
6. Kucherov, D.P. Modeling traffic in control problems of robotic systems. In: Proceedings the Fifth world congress “Aviation in XXI-st century”. 2012. P. 18 – 19.
7. Huang, Qiwang, et al. Cooperative searching strategy for multiple unmanned aerial vehicles based on modified probability map. In: Theory, Methodology, Tools and Applications for Modeling and Simulation of Complex Systems: 16th Asia Simulation Conference and SCS Autumn Simulation Multi-Conference, AsiaSim/SCS AutumnSim 2016, Beijing, China, October 8-11, 2016. P. 279 – 287.
8. Teichmann, Dušan, et al. Unmanned aerial vehicles routing problem. In: Proceedings of the 2014 15th International Carpathian Control Conference (ICCC). IEEE, 2014. P. 602 – 07.
9. Pitre, Ryan R.; Li, X. Rong; Delbalzo, R. UAV route planning for joint search and track missions. IEEE Transactions on Aerospace and Electronic Systems, 2012. P. 2551 – 2565.
10. Ha, Il-Kyu; Cho, You-Ze. A probabilistic target search algorithm based on hierarchical collaboration for improving rapidity of drones. Sensors, 2018. P. 25 – 35.
11. Lee, Min-Hyuck; Yeom, Seokwon. Detection and tracking of multiple moving vehicles with a UAV. International Journal of Fuzzy Logic and Intelligent Systems, 2018. P. 182 – 189.
12. Dakhno N., Barabash O., Shevchenko H., Leshchenko O., Dudnik A. Integro-differential Models with a K-symmetric Operator for Controlling Unmanned Aerial Vehicles Using a Improved Gradient Method. 2021 IEEE 6th International Conference “Actual Problems of Unmanned Aerial Vehicles Development (APUAVD). Proceedings. October 19 – 21, 2021, Kyiv, Ukraine. P. 61 – 65.
13. Barabash O., Dakhno N., Shevchenko H., Sobchuk V. Unmanned Aerial Vehicles Flight Trajectory Optimisation on the Basis of Variational Enequality Algorithm and Projection Method. Proceeding. 2019 IEEE 5th International Conference “Actual Problems of Unmanned Aerial Vehicles Developments” (APUAVD). 22-24 October, National Aviation University, 2019. Kyiv, Ukraine. P. 136 – 139.