МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет радіоелектроніки
Факультет Комп'ютерних наук
Кафедра Програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

_____другий (магістерський)_____
(рівень вищої освіти)

Дослідження методів мурашиних колоній для вирішення задачі QSAR

Виконав:
студент __2__ курсу групи ____ІПЗм-20-1_____
_____Коротач І.В._____
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного
_____забезпечення_____
Тип програми _____Освітньо-наукова_____
Керівник _____доц. Лещинський В.О._____
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар

2022 р.

Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)

Кафедра _____Програмної інженерії_____
(повна назва)

Рівень вищої освіти_____другий (магістерський)_____

Спеціальність _____121 – Інженерія програмного забезпечення_____
(код і повна назва спеціальності)

Тип програми _____освітньо-наукова програма_____
(освітньо-професійна або освітньо-наукова)

Освітня програма _____Інженерія програмного забезпечення_____
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____20 ____ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студента _____*Коротача Ігоря Вячеславовича*_____
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження методів мурашиних колоній для вирішення задачі QSAR» _____

затверджена наказом університету від «24» березня 2022 р. № 412 Ст

2. Термін подання роботи до екзаменаційної комісії «__» _____ 2022 р.

3. Вихідні дані до роботи *методи вирішення задачі QSAR, критерії їх ефективності, еталонна реалізація алгоритму мурашиних колоній.*

4. Перелік питань, що потрібно опрацювати в роботі *вступ, аналіз предметної області, аналіз методів вирішення задачі QSAR, аналіз алгоритмів мурашиних колоній, критерії їх оцінки, постановка задачі, створення авторського алгоритму, аналіз його результатів на задачі QSAR, формування висновків.*

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної області	25.02.2022	виконано
2	Постановка задачі	10.03.2022	виконано
3	Проведення дослідження	01.04.2022	виконано
4	Підготовка пояснювальної записки	03.05.2022	виконано
5	Підготовка презентації та доповіді	06.05.2022	виконано
6	Попередній захист	10.05.2022	виконано
7	Перевірка на академічний плагіат	10.05.2022	виконано
8	Нормоконтроль	12.05.2022	виконано
9	Рецензування	13.05.2022	виконано
10	Занесення диплома в електронний архів	15.05.2022	виконано
11	Допуск до захисту у зав. кафедри	15.05.2022	виконано

Дата видачі завдання 17.01.2022 р.

Студент _____
(підпис)

Керівник роботи_____ доцент Лещинський В.О.
(підпис)

РЕФЕРАТ / ABSTRACT

Кваліфікаційна робота магістра містить: 58 стор., 10 рис., 3 табл., 8 джерел, 2 додатки.

Об’єктом дослідження є задача QSAR та оптимізації хімічних завдань за допомогою методу оптимізації мурашиних колоній.

Метою дослідження є аналіз задачі QSAR, її етапів, методів її вирішення. Крім цього, метою можна вважати дослідження використання методу оптимізаціїї мурашиних колоній, спектрів його використання у хемоінформатиці.

У результаті роботи проведено аналіз задачі QSAR, був зроблений огляд найбільш популярних методів її вирішення та експериментальне порівняння з методом оптимізації мурашиних колоній.

ВИВЧЕННЯ, ЛІКИ, КОНСТРУЮВАННЯ, ОБРАННЯ ОЗНАК, ГЕНЕТИЧНИЙ АЛГОРИТМ, ДАТА МАЙНІНГ, МУРАШИНИЙ АЛГОРИТМ.

The object of research is the solution of QSAR and other chemoinformatic problems with the help of the Ant Colony Optimization algorithm.

The aim of the research is the analysis of QSAR, its steps and possible solution strategies. Moreover, the applicability of Ant Colony Optimization as optimization algorithm in chemistry is considered.

As a result of research practice, the analysis of QSAR and its most popular solutions was carried out. The feasibility and performance comparison of Ant Colony Optimizations and alternatives was undertaken.

DISCOVERY, DRUG, DESIGN, FEATURE SELECTION, GENETIC ALGORITHM, QSAR, ANT COLONY OPTIMIZATION.

Я, Коротач Ігор Вячеславович, студент групи ІПЗм-20-1, здобувач вищої освіти на другому (магістерському) рівні, кафедра Програмної інженерії, заявляю: моя кваліфікаційна робота на тему «Дослідження методів мурашиних колоній для вирішення задачі QSAR», що буде представлена до ЕК для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлена з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

QSAR	Quantitative structure–activity relationship, метод хемоінформатики
SAR	Structure–activity relationship, метод хемоінформатики
$\boldsymbol{LD_{50}}$	Lethal dose 50, смертельна доза при якій 50% досліджуваних об’єктів вмирають
ACO	Ant Colony Optimization, алгоритм оптимізації мурашиних колоній
RFID	Radio frequency identification, радіочастотна ідентифікаційна мітка
GPU	Graphics Processing Unit, окремий пристрій персонального комп'ютера або ігрової приставки, виконує графічний рендеринг

ВСТУП

В останні роки зростає потреба в нових методологіях видобутку даних, які можуть аналізувати та інтерпретувати великі обсяги даних. Методи штучного інтелекту, такі як штучні нейронні мережі, дерева класифікації та регресії, k-nearest neighbor алгоритми широко використовувались для аналізу та кореляції хімічних та біологічні дані [1]. Багато з цих методів використовуються разом з оптимізаційними техніками, включаючи різні жадібні алгоритми, а також стохастичну оптимізацію такі підходи, як імітація відпалу та генетичні алгоритми.

Особливу увагу цим методам надають сфери, які шукають нові, або не виявлені залежності у великих базах даних властивостей сполук.

Однією з таких сфер є QSAR. Одним із перших історичних застосувань QSAR було передбачення температур кипіння. Іншими сферами можуть бути молекулярне моделювання та докінг.

Основним припущенням для всієї хімічної молекулярної гіпотези є те, що подібні молекули мають подібну активність. Цей принцип також називають структурно-активними зв'язками (SAR). Основна проблема полягає в тому, як визначити невелику різницю на молекулярному рівні, оскільки кожен вид діяльності, наприклад здатність до реакції, здатність до біотрансформації, розчинність, цільова активність тощо можуть залежати від іншої різниці.

Створені гіпотези зазвичай покладаються на кінцеве число хімічних даних. Таким чином, принцип індукції слід поважати, щоб уникнути надмірних гіпотез та виникати надмірне та марне трактування структурних / молекулярних даних. Парадокс SAR свідчить про те, що всі подібні молекули не мають подібних дій [2].

Загалом, процес SAR/QSAR здебільшого поділений на декілька основних етапів: вибір набору даних та вилучення структурних / емпіричних дескрипторів, змінний відбір побудованої моделі та перевірка оцінки. Дана робота фокусується на останніх трьох.

Метою цієї кваліфікаційної роботи є кроки побудови моделі та вилучення нерелевантних дескрипторів. У ході роботи роботи будуть розглянуті різні методи та стратегії feature selection, та найбільша увага приділяється перспективам застосування підходу оптимізації мурашиних колоній для вирішення хімічних завдань. Під час огляду представлені модифікації алгоритму, їх аналітичний огляд та практичне застосування для вирішення завдання QSAR та аналіз придатності для роботи зі схожими проблемами хімічної інформатики сучасності.

Особливу актуальність набуває сфера комп’ютерного конструювання ліків після випадків мутації вірусів типу COVID, що потребує прискореного темпу створення нових вакцин та препаратів. В ідеалі, обчислювальний метод у змозі передбачити спорідненість ще перед тим як сполука буде синтезована, а отже, теоретично, лише одну речовину доведеться синтезувати, економлячи величезну кількість часу та коштів. Реальність полягає в тому, що сучасні обчислювальні методи недосконалі і забезпечують, у кращому випадку, лише якісно точну оцінку спорідненості. На практиці ж і досі проводять численні повторення конструювання, синтезу та тестування, перш ніж відкрити оптимальний препарат. Обчислювальні методи прискорюють розробку шляхом зменшення кількості необхідних повторень і часто передбачують нові структури.

Також широке застосування та розвиток сфер, які використовують методи комп'ютерної хімії надасть можливість зменшити залежність від випадкового характеру відкриттів та перейти до більш прогнозованих та кількісно обчислювальних методів.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Алгоритм оптимізації мурашиних колоній

В природі, мурахи (початково) блукають довільним чином, і по знаходженні їжі повертаються до колонії, залишаючи по собі феромонний слід. Якщо інші мурахи знаходять такий шлях, вони схильні припинити свої блукання, натомість слідувати позначеним шляхом, посилюючи його під час повернення у разі знайдення їжі.

Однак, з часом, феромонові шляхи випаровуються, тоді привабливість шляхів зменшується. Чим більше часу потрібно мурасі, щоб подолати дорогу, тим більше часу мають феромони, щоб випаруватись. Натомість, короткий шлях проходиться частіше, отже щільність феромонів стає більшою на короткому шляху. Випаровування феромонів також надає перевагу уникнення локально найкращих шляхів. Якби випаровування не відбувалось взагалі, шляхи обрані першим мурахою тяжіли б стати вкрай привабливими для наступних. В цьому разі, розвідка можливих шляхів була б обмежена.

Таким чином, коли мураха знаходить вдалий (тобто короткий) шлях з колонії до джерела їжі, інші мурахи швидше слідуватимуть йому, і позитивний зворотний зв'язок зрештою призведе до обрання цього шляху всіма мурахами. Ідея мурашиного алгоритму полягає в наслідуванні поведінки з «симулятором мурахи», що прогулюється графом, який представляє проблему, що треба розв'язати [3].

1.2. Прийняття рішень

Первісна ідея прийшла зі спостереження за використанням харчових ресурсів серед мурах, де мурахи, окремо обмежені в своїх пізнавальних

можливостях, колективно здатні знайти найкоротший шлях між джерелом їжі і гніздом. На рисунку 1.1.1 схематично зображена поведінка мурах

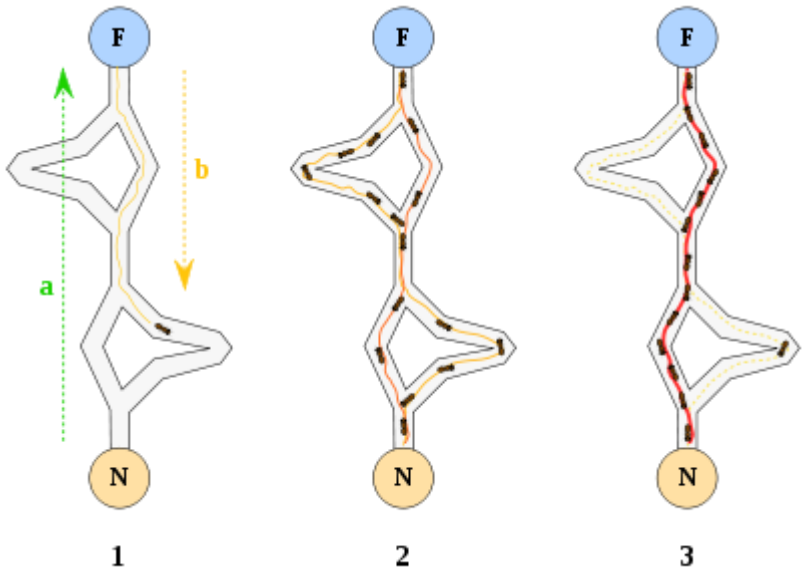

Рис. 1.2.1 Прокладення шляху мурахами

1. Перша мураха знаходить джерело їжі (F), через якийсь шлях (a), тоді повертається до гнізда (N), залишивши позаду слід з феромонів (б)
2. Мурахи без розбору обирають всі чотири шляхи, але підсилення основної стежки робить її привабливішою як найкоротший шлях.
3. Мурахи обирають коротший шлях, довгі відтинки втрачають щільність феромонового сліду.

В серії дослідів на колонії мурах з вибором між двома шляхами різної довжини, які ведуть до джерела їжі, біологи спостерігали, що мурахи тяжіють до використання найкоротшого шляху. Модель, що пояснює таку поведінку така:

1. Мураха (звана «бліц») рухається більш-менш випадково по колонії;
2. Якщо вона знаходить джерело їжі, вона повертається прямо до гнізда, залишаючи по собі феромоновий слід;
3. Ці феромони заманливі, ближні мурахи схилятимуться до слідування, більш чи менш точно, цим шляхом;
4. Повертаючись до колонії, мурахи підсилюватимуть маршрут;

5. Якщо наявні два шляхи до одного й того самого джерела їжі тоді, за певний час, коротший шлях пройде більше мурах ніж довшій;

6. Короткий шлях все більш посилюватиметься, і таким чином ставатиме привабливішим;

7. Довгий шлях з часом зникне, бо феромони вивітряться;

8. Зрештою, всі мурахи визначатимуть і через це обиратимуть найкоротший шлях.

Мурахи використовують навколишнє середовище як посередник для зв'язку. Вони обмінюються інформацією непрямо через відкладання феромонів, які уточнюють статус їхньої роботи. Інформація розповсюджувана через феромони має місцеву дію, лише мурахи розташовані поруч із відкладеними феромонами помічають їх. Таку систему називають «Стіґмерґі» і вона трапляється в багатьох спільнотах соціальних тварин (її вивчали на прикладі розбудови стовпів у гніздах термітів) [3]. Спільне розв'язання проблем занадто складних для одного мурахи є хорошим прикладом самоорганізованої системи. Система покладається на позитивний зворотний зв'язок (відкладення феромонів приваблюють інших мурах, які підсилять їх у свою чергу) і негативний (зникнення маршруту через випаровування забезпечує оптимальність роботи системи). Теоретично, якщо щільність феромонів залишатиметься постійною на всіх відтинках, жоден із шляхів не стане головним. Однак, через зворотний зв'язок, малі відмінності на підмаршрутах підсилюватимуться і дозволятимуть зробити вибір. Алгоритм зсуватиметься з нестабільного стану, де жоден з підмаршрутів не привабливіший за інші, у бік стабільного стану, де маршрут утворений з найсильніших підмаршрутів.

1.3. Опис алгоритму

Припустимо, що навколишнє середовище для мурах представляє повнозв'язний неорієнтований граф. Кожне ребро має вагу, яка позначається як відстань між двома вершинами, що ним з'єднується. Граф є двохскерованим, тому мураха може подорожувати по грані в будь-якому напрямку.

Ймовірність включення ребра в маршрут окремої мурахи пропорційна до кількості феромонів на цьому ребрі, а кількість відкладеного феромону пропорційне до довжини маршруту. Чим коротший маршрут, тим більше феромону буде відкладено на його ребрах, отже, більша кількість мурах буде включати його в синтез власних маршрутів. Моделювання такого підходу, що використовує тільки додатній зворотний зв'язок, призводить до передчасної збіжності — більшість мурашок рухається по локально-оптимальному маршруту.

Уникнути цього можна моделюючи від'ємний зворотний зв'язок у вигляді випаровування феромону. Причому, якщо феромон випаровується швидко, то це призводить до втрати пам'яті колонії і забування хороших рішень, з іншого боку, збільшення часу випарів може призвести до отримання стійкого локального оптимального рішення.

Пройдений мурахою шлях відображається, коли мураха відвідає всі вузли графа. Цикли заборонено, оскільки в алгоритм включено список табу. Після завершення довжина шляху може бути підрахована — вона дорівнює сумі довжин всіх ребер, якими подорожувала мураха. Рівняння (1) показує кількість феромону, який був залишений на кожному ребрі шляху для мурашки k. Змінна Q є константою.

$$\triangle \tau_{ij}^{k}(t) = \frac{Q}{L^{k}(t)} \quad (1)$$

Результат рівняння є засобом вимірювання шляху, — короткий шлях характеризується високою концентрацією феромонів, а більш довгий шлях — більш низькою. Далі, отриманий результат використовується в рівнянні (2), щоб збільшити кількість феромону вздовж кожного ребра пройденого мурахою шляху.

$$\tau_{ij}(t) =\triangle \tau_{ij}(t) + (\tau_{ij}^{k} \times \rho) \quad (2)$$

Важливо, що дане рівняння застосовується до всього шляху, при цьому кожне ребро позначається феромоном пропорційне до довжини шляху. Тому слід дочекатися, поки мураха закінчить подорож і лише потім оновити рівні феромону, в іншому випадку справжня довжина шляху залишиться невідомою. Константа p — значення між 0 і 1.

На початку шляху у кожного ребра є шанс бути обраним. Щоб поступово видалити ребра, які входять в гірші шляхи графа, до всіх ребер застосовується процедура випаровування феромону. Використовуючи константу p з рівняння (2), отримуємо рівняння (3):

$$\tau_{ij}(t) = \tau_{ij}(t) \times (1 - \rho) \quad (3)$$

Для випаровування феромону використовується зворотний коефіцієнт оновлення шляху.

1.4. Еталонна реалізація

Програмна реалізація алгоритму може бути проілюстрована за допомогою наступного псевдокоду:

```
procedure ACO_MetaHeuristic is
  while not terminated do
    generateSolutions()
    daemonActions()
    pheromoneUpdate()
  repeat
end procedure
```

Булевим значенням «terminated» може бути як досягнення необхідної кількості кроків, або знаходження путі, яка задовольняє необхідний критерій результату.

Функція «generateSolutions» проектується на кожну мураху таким чином, що кожна мураха самостійно обходить граф та записує свою путь.

Функція «daemonActions» є агрегаційною та відповідає за порівння результатів моделі (путі) кожної мурахи та знайдення найліпшого варіанту.

Функція «pheromoneUpdate» оновлює показники феромону на кожному участку путі.

1.5. Пошук кількісних співвідношень структура-властивість (QSAR)

QSAR – це методологія, яка використовує математичні моделі для кореляції біологічної активності (наприклад, LD_{50}) та описових параметрів (дескрипторів), пов’язаних із структурою молекули. QSAR – це регресійна модель, яка широко застосовуються в біологічних та хімічних науках. У галузі класифікації небезпечних хімічних речовин моделі QSAR можуть бути корисними для прогнозування природи різних хімічних речовин у випадках, коли для цієї мети немає експериментальних даних.

Основне припущення для дослідження QSAR полягає в тому, що молекули, які мають подібну структуру, як правило, мають подібні властивості. Мета моделі QSAR полягає в тому, щоб зв'язати групу змінних-провісників з потужністю змінної, що реагує. Провісники містять інформацію, яка описує теоретичні молекулярні дескриптори або фізико-хімічні властивості хімічних речовин; чутливі змінні включають різні властивості, такі як біологічна активність, які використовуються для класифікації небезпечних хімічних речовин. Розробка дескрипторів на основі молекулярної структури та кореляція молекулярних дескрипторів із чутливою активністю за допомогою багатофакторного аналізу є двома кроками для створення моделей QSAR. У цій главі не розглядається процес розробки моделей QSAR.

Провісники містять хімічну інформацію, закодовану в молекулярній структурі з практичної точки зору, а хімічна інформація включає конституційні, геометричні, гідрофобні, електронні, стеричні та топологічні дескриптори.

Під час розробки моделей QSAR можна зазначити, що розглядаючи всі обчислені дескриптори в моделі, можна отримати дуже високий коефіцієнт кореляції, але в той же час модель стає настільки складною, що неможливо інтерпретувати результат. Отже, метою має бути відбір значущих дескрипторів шляхом усунення факторів мультиколінеарності та випадкової кореляції. Вибір ознак контролює це завдання, яке є важливою основою для побудови моделі.

2. АНАЛІЗ ІСНУЮЧИХ АЛГОРИТМІВ ACO

Перший алгоритм ACO, званий Ant System (AS), був застосований до задачі комівояжера (Travelling Salesman Problem – TSP). Це дало обнадійливі результати, проте його ефективність не конкурувала з найсучаснішими алгоритмами для TSP. Отже, одним із важливих напрямків досліджень алгоритмів ACO було впровадження алгоритмічні вдосконалення для досягнення набагато кращої продуктивності. Як правило, ці вдосконалені алгоритми були знову протестовані на TSP. Хоча вони відрізняються головним чином у конкретних аспектах управління пошуком, усі ці алгоритми ACO є заснований на більш сильній експлуатації історії пошуку для спрямування процесу пошуку мурах. Нещодавні дослідження характеристик простору пошуку деяких комбінаторних проблеми оптимізації показали, що для багатьох проблем існує кореляція між якістю рішення та відстанню від дуже хороших або оптимальних рішень. Отже, представляється розумним припустити, що концентрація пошук найкращих рішень, знайдених під час пошуку, є ключовим аспектом, який привів до покращеної продуктивності, показаної модифікованими алгоритмами ACO.

2.1. Варіація Max-min ant system (MMAS)

Ключовим для досягнення найкращої продуктивності алгоритмів ACO є поєднання вдосконаленого використання найкращих рішень, знайдених під час пошуку, з ефективним механізмом уникнення ранньої застою пошуку. Система мурашок MAX –MIN, яка була спеціально розроблена для задоволення цих вимог, відрізняється від AS в трьох ключових аспектах [5]:

а) Використовувати найкращі рішення, знайдені під час ітерації або під час запуску алгоритм, після кожної ітерації лише один мураха додає феромон. Цей

мураха може бути тим, хто знайшов найкраще рішення в поточній ітерації (iteration-best) або той, який знайшов найкраще рішення з початку алгоритму (global-best).

б) Щоб уникнути застою пошуку, діапазон можливих феромонових стежок на кожному компонент розчину обмежений інтервалом $[\tau_{min}; \tau_{max}]$.

в) Крім того, ми навмисно ініціалізуємо феромонові стежки до максимуму, досягаючи таким чином більш високе дослідження рішень на початку алгоритму.

2.2. Варіація Elitist ant system

Якість рішень, вироблених мурашиною системою, можна було б покращити за допомогою так званих елітарних мурах. Ідея елітарної стратегії в контексті мурашиної системи полягає в тому, щоб підкреслити найкращий шлях, знайдений після кожної ітерації. Коли оновлюються маршрути цей шлях трактується так, ніби певна кількість мурах, а саме елітарних мурах, обрала цей шлях. Оскільки цілком ймовірно, що деякі краї цього шляху є частиною оптимального рішення, мета полягає в тому, щоб керувати пошуком у наступних ітераціях [5].

2.3. Варіація Rank-based ant system (ASRank)

Концепція ранжування може бути застосована і поширена на мурашину систему наступним чином: після того, як усі мурахи сформували тур, мурахи сортуються за тривалістю туру ($L_1 < L_2 < ... < L_m$), а внесок мурахи в оновлення рівня сліду зважується відповідно до рангу μ мурахи. Крім того, враховуються лише ω найкращих мурах. Таким чином, можна уникнути небезпеки надмірно

підкреслених феромонових стежок, спричинених багатьма мурахами, які використовують неоптимальні шляхи. Оскільки σ – вага внеску на найкращий тур, знайдений на поточній ітерації не слід перевищувати будь-якою іншою вагою. З іншого боку, розумним є ідея використовувати "один" як мінімальну вагу. З цієї причини використовується вага (σ - μ) для μ -го найкращого мурахи і встановити ω = σ - 1, що означає, що кількість розглянутих мурах перевищує кількість елітарних мурах на одного.

2.4. Варіація Continious orthogonal ant colony (COAC)

Метод ортогонального проектування є способом планування експериментів і він широко застосовується в багатофакторних експериментах. В експерименті параметри називаються факторами, тоді як значення цих параметрів – рівнями. Розглянемо експеримент з *k*-факторами і кожен фактор з рівнями *s*. Тоді будуть тестові комбінації s^k. Це повномасштабний експеримент, і вартість обчислення експоненціально зростає, коли *k* і *s* стають великими. Для того, щоб зменшити кількість експериментів, щоб зробити проблему відстежуваною, використовується метод ортогонального проектування

Мурахи, яких відправляють, щоб знайти оптимальне місце в даному домені, використовують феромон і ортогональне дослідження для виконання місії. Домен розділений на кілька регіонів різного розміру. Кожна область має багато властивостей: радіуси пошуку, координати центру, кількість феромону та ранги за своєю бажаністю. Бажаність оцінюється цільовою функцією.

2.5 Перспективи та складності застосування

Деякі ситуації потребують нових концепцій, оскільки «інтелект» часто не є централізованим, а знаходиться у найменших об'єктах. Антропоцентричні концепції завжди приводить нас до виробництва IT-систем, в яких використовується централізована обробка даних, блоки управління та обчислювальні потужності. Такі централізовані підрозділи постійно підвищують свою продуктивність і можуть порівнюватись з людським мозком. Модель мозку стала еталонним баченням комп'ютерів.

Невеликі пристрої, які можна порівняти з комахами, не мають власного розвинутого інтелекту. Навпаки, їх інтелект можна назвати досить обмеженим. Наприклад, неможливо інтегрувати високоефективний калькулятор з можливістю вирішувати будь-яку математичну задачу в біочіп, який імплантується в організм людини. Однак, коли об'єкти взаємопов'язані, вони розпоряджаються формою інтелекту, яку можна порівняти з колонією мурах або бджіл. Для певних задач цей тип інтелекту може навіть перевершувати очікування про централізовану систему, подібну до мозку [4].

Природа дала нам кілька прикладів того, як мізерні організми, якщо всі вони слідують одному і тому ж основному правилу, можуть створити форму колективного інтелекту на макроскопічному рівні. Колонії соціальних комах чудово ілюструють цю модель, яка сильно відрізняється від людських суспільств. Ця модель базується на взаємодії незалежних підрозділів з простою та непередбачуваною поведінкою.

Зв'язок на основі феромонів є одним з найбільш ефективних способів спілкування, який широко спостерігається в природі. Феромон використовується соціальними комахами, такими як бджоли, мурахи і терміти. У зв'язку з його доцільністю штучні феромони були прийняті в системах мультироботів і робототехнічних систем зі структурою роя. Зв'язок на основі феромонів був реалізований різними засобами, такими як хімічний або фізичний (RFID-мітки,

світло, звук) способи. Проте, ці реалізації не змогли повторити всі аспекти феромонів, які використовуються в живій природі.

3. ПОСТАНОВКА ЗАВДАННЯ

У кваліфікаційній роботі необхідно проаналізувати алгоритми оптимізації мурашиної колонії для вирішення задачі QSAR та інших завдань комп'ютерної хімії, вказати на різницю у формі структури вхідних значень для різних задач та зробити висновок про найбільш оптимізований варіант на основі наступних критеріїв:

а) здатність до паралелізації
б) швидкість роботи на одному вузлі
в) гнучкість зміни
г) повторюваність результатів
д) складність алгоритму

Порівняння алгоритмів проводяться аналітично. Крім того, необхідно проаналізувати альтернативні алгоритми з урахуванням інших підходів до вирішення завдання QSAR. Метою кваліфікаційної роботи є створення авторського гібридного алгоритму, що поєднує найбільш значущі переваги аналогів. Експериментальна частина формується на основі порівняння авторського алгоритму з альтернативними реалізаціями, не заснованими на стратегії. Для цього здійснюється пошук за 2,3,4-дескрипторним простором у заздалегідь програмно згенерованому спектрі хімічних дескрипторів та верифікація за допомогою повного перебору 2 та 3-дескрипторного простору. Як висновок необхідно подати результати дослідження та експериментів та рекомендації щодо подальшого поліпшення.

4. АЛГОРИТМИ ВИРІШЕННЯ ЗАДАЧІ QSAR

4.1 Генетичний алгоритм (GA)

Генетичний алгоритм — це евристичний метод пошуку, який імітує процес природного відбору. Там, де вичерпний пошук недоцільний, використовуються евристичні методи для прискорення процесу пошуку задовільного рішення. Генетичні алгоритми належать до більшого класу еволюційних алгоритмів (EA), які генерують рішення проблем оптимізації за допомогою методів, натхненних природною еволюцією, таких як успадкування, кросовер, мутація та відбір. Еволюція зазвичай починається з популяції випадково згенерованих індивідів і є ітераційним процесом, при цьому популяція на кожній ітерації відома як покоління. У кожному поколінні оцінюється придатність кожної особини в популяції; придатність – це зазвичай значення цільової функції в оптимізаційній задачі, що вирішується. Більш придатні особини стохастично відбираються з поточної популяції, і геном кожної особи модифікується (рекомбінується і, можливо, випадково мутується) для формування нового покоління [6]. Нове покоління варіантів рішень потім використовується на наступній ітерації алгоритму. Зазвичай алгоритм припиняє роботу, коли або створено максимальну кількість поколінь, або досягнуто задовільного рівня придатності для популяції.

Зазвичай, генетичні алгоритми мають наступну схему етапів (рис. 4.1.1):

а) Ініціалізація генів

б) Відбір покоління

в) Накладання генетичних операторів

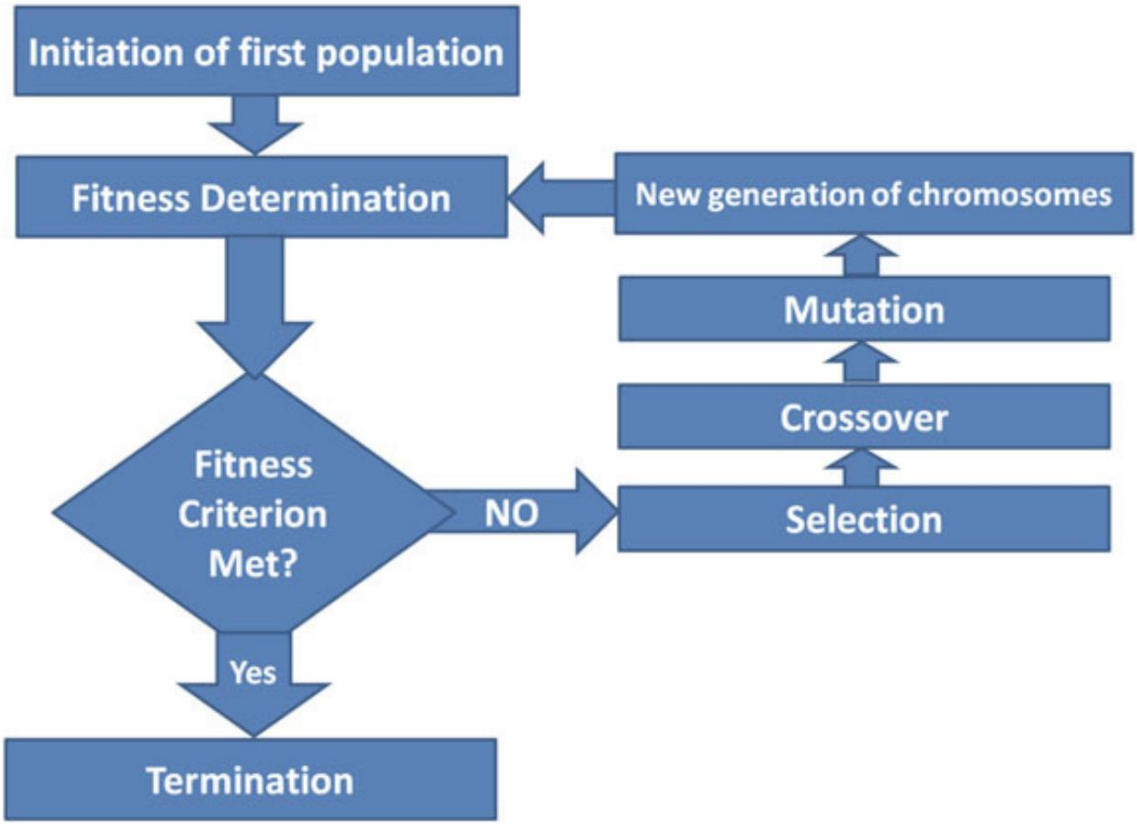

Рис 4.1.1 - Схема етапів Genetic Algorithm

Під час ініціалізації багато індивідуальних рішень зазвичай генеруються випадковим чином, щоб утворити початкову сукупність, що дозволяє отримати весь діапазон можливих рішень. Іноді рішення можуть бути «засіяні» в областях, де, ймовірно, будуть знайдені оптимальні рішення.

Протягом кожного наступного покоління відбирається частка існуючої популяції для виведення нового покоління. Індивідуальні рішення вибираються за допомогою функції фітнесу, яка оцінює кожну особистість і на основі цієї функції фітнесу відбирає найкращих індивідів. Фітнес функція генетичного алгоритму для вирішення QSAR виглядає наступним чином:

$$F = \sum_{i=0}^{i=k} \frac{P_i - \hat{P}_i}{\hat{P}_{i,Max} - \hat{P}_{i,Min}}$$

де F – значення фітнес функції,

k – кількість параметрів,

P_i – значення властивості у поточному гені

$\hat{P}$ – бажане значення властивості

$\hat{P}_{i,Max}$ – бажане максимальне значення властивості

$\hat{P}_{i,Min}$ – бажане мінімальне значення властивості

Наступним кроком є створення популяції другого покоління рішень із тих, що вибрано за допомогою комбінації генетичних операторів: кросовер (також званий як рекомбінація) та мутація.

Для кожного нового розчину, який буде виготовлено, вибирається пара «батьківських» розчинів для розведення з попередньо вибраного пулу. Створюючи «дитяче» рішення з використанням вищевказаних методів схрещування та мутації, створюється нове рішення, яке, як правило, поділяє багато характеристик своїх «батьків» [6]. Як правило, середня пристосованість підвищується завдяки цій процедурі для популяції, оскільки лише для розведення відбираються найкращі рішення з першого покоління, а також невелика частка менш підходящих рішень. Ці менш відповідні рішення забезпечують генетичне різноманіття в генетичному фонді батьків і, отже, забезпечують генетичну різноманітність наступного покоління дітей (див. рис. 4.1.2). Хоча кросовер і мутація відомі як основні генетичні оператори, у генетичних алгоритмах можна використовувати інші оператори, такі як перегрупування, колонізація-вимирання або міграція. Варто налаштувати такі параметри, як ймовірність мутації, ймовірність кросинговеру та розмір популяції, щоб знайти розумні налаштування для класу проблеми, над яким працюється.

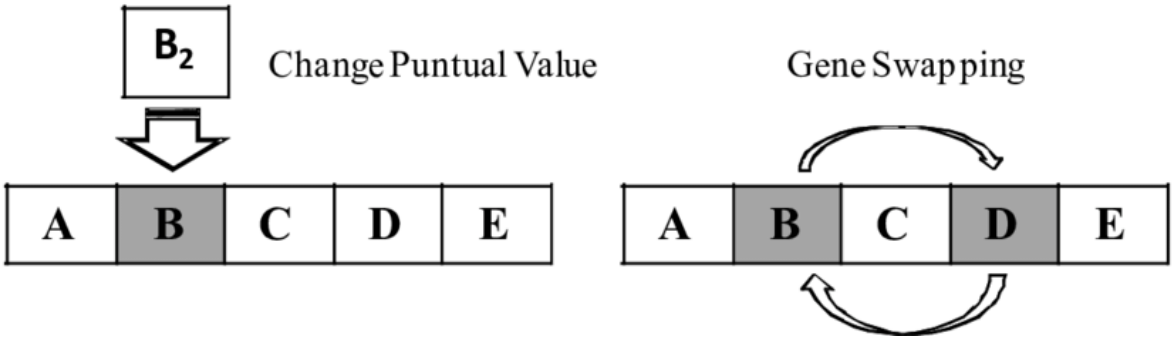

Рис. 4.1.2 Операції кросоверу та мутації

Цей процес повторюється, поки не буде досягнуто умови припинення. Умови припинення:

а) Знайдено рішення, яке задовольняє мінімальним критеріям.

б) Досягається фіксована кількість поколінь (зазвичай визначається користувачем).

В результаті роботи алгоритму обирається формула, яка дає сукупність дескрипторів, що задовольняють умові шуканої властивості. Алгоритм є дуже гнучким та добре паралізується. Мінусами є схильність до локальних максимумів та велика кількість конфігураційних параметрів.

4.2Алгоритм імітації відпалу

Алгоритм імітації відпалу — це загальний алгоритм стохастичного пошуку, створений на основі процесу, який використовується в металургії. Техніка нагрівання та повільного охолодження відпалу дозволяє спочатку збудженим і дезорганізованим атомам металу знайти міцні, стабільні конфігурації. Аналогічно імітований відпал шукає рішення проблем оптимізації, спочатку маніпулюючи рішенням випадковим чином (висока температура), а потім повільно збільшуючи коефіцієнт жадібних покращень (охолодження), доки не буде знайдено подальших покращень. Вибір ознак — це процес, який вибирає підмножину вихідних ознак.

Оптимальність підмножини ознак вимірюється за критерієм оцінки. Зі збільшенням розмірності домену кількість ознак N збільшується [7]. Типовий процес вибору ознак складається з чотирьох основних кроків, а саме:

а) створення підмножини
б) оцінка підмножини
в) критерій зупинки
г) перевірка результату.

Генерація підмножини — це процедура пошуку, яка створює підмножини можливостей-кандидатів для оцінки на основі певної стратегії пошуку. Кожна підмножина кандидатів оцінюється та порівнюється з попередньою найкращою відповідно до певного критерію оцінки. Якщо нова підмножина виявляється кращою, вона замінює попередню найкращу підмножину. Процес генерації та оцінки підмножини повторюється, поки не буде задоволено заданий критерій зупинки. Тоді вибрану найкращу підмножину зазвичай потрібно підтвердити попередніми знаннями або різними тестами за допомогою синтетичних і реальних наборів даних [7].

Загалом, основною частиною алгоритму є обрання трьох основних параметрів:

а) Annealing schedule – параметр шуканої величини та часу відпалу
б) Фітнес функції – функції оцінки результатів відпалу
в) Simulated annealing – функції, яка маючи вхідне рішення та температуру переходить до сусіднього рішення

Роль температури – регулювати розмір околиці. При високій температурі околиці повинні бути великими, що дозволить алгоритму широко досліджувати. При низькій температурі околиці повинні бути невеликими, що змушує алгоритм досліджувати локально. Наприклад, якщо ми представляємо набір доступних функцій у вигляді бітового вектора, так що кожен біт вказує на наявність або відсутність певної функції. Цей алгоритм намагається ітеративно покращити випадково згенероване початкове рішення. На кожній ітерації алгоритм генерує сусіднє рішення і обчислює різницю в якості (енергії, за аналогією з металургійним

процесом) між поточним і потенційним рішеннями. Якщо нове рішення краще, то воно зберігається (рис. 4.2.1). У ситуації вирішення задачі QSAR параметрами алгоритму обираються:

а) Дескриптори як елементи бітового вектору

б) Значення функції фітнесу за базою моделі, як Multiple Linear Regression

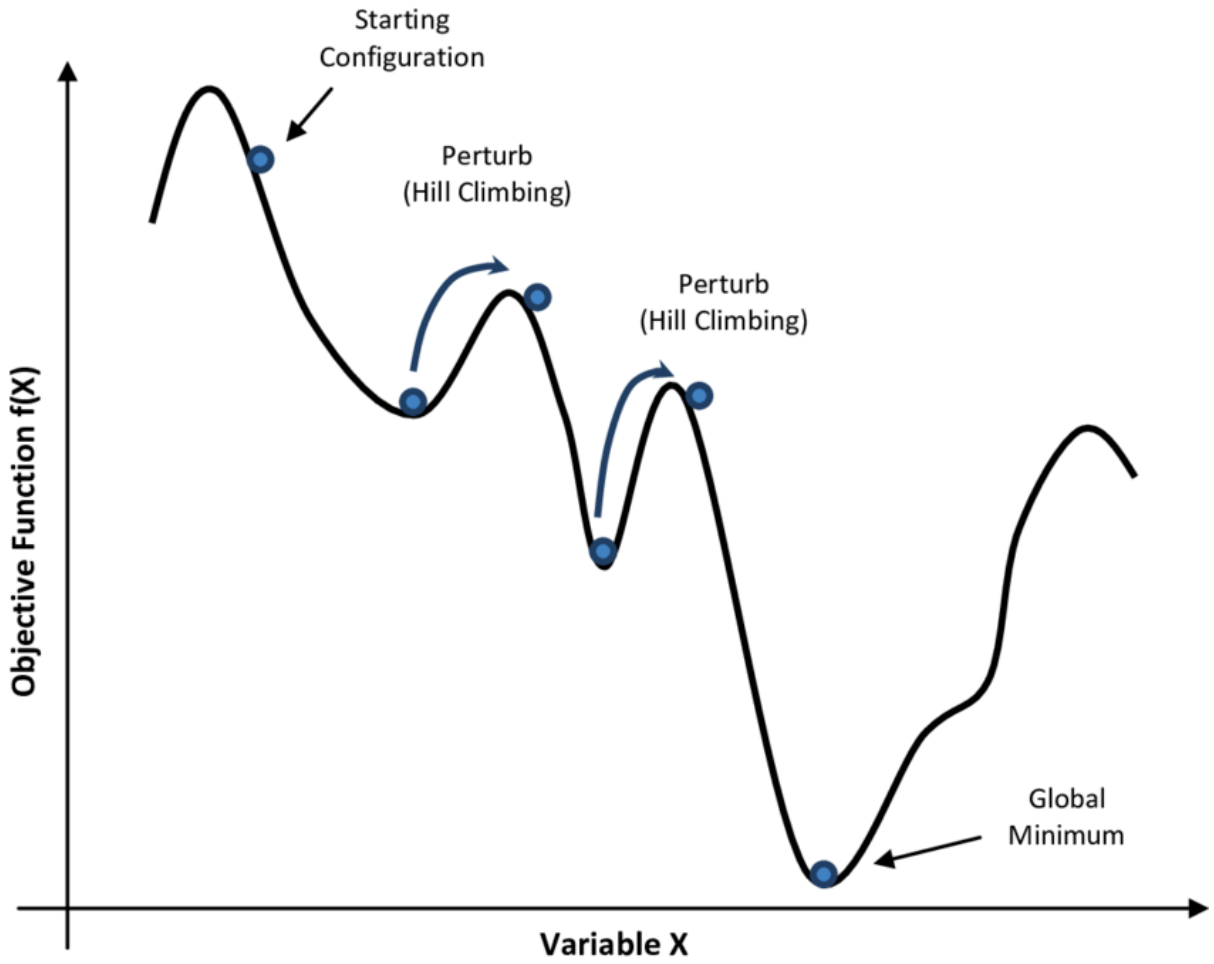

Рис. 4.2.1 Ілюстрація алгоритму відпалу

Зазвичай, функція simulated annealing є ключовою для пошуку оптимальних параметрів алгоритму, її форма слідує наступній формі:

$$P = \begin{cases} 1 & if \Delta c \leq 0 \\ e^{-\Delta c / t} & if \Delta c > 0 \end{cases}$$

де P – ймовірність обрання нового рішення,

Δc – різниця у помилці рішення,

t – поточна температура

Таким чином, алгоритм імітації відпалу дуже ефективний в ситуаціях великої кількості локальних екстремумів та дає найбільш точний результат, але є більш вимогливим до використання ресурсів та потребує більше часу.

4.3 Штучні нейронні мережі

Штучні нейронні мережі (Neural Networks) — це обчислювальні системи, натхнені біологічними нейронними мережами, що складають мозок тварин. Такі системи навчаються задач (поступально покращують свою продуктивність на них), розглядаючи приклади, загалом без спеціального програмування під задачу. Наприклад, у розпізнаванні зображень вони можуть навчатися ідентифікувати зображення, які містять котів, аналізуючи приклади зображень, мічені як «кіт» і «не кіт», і використовуючи результати для ідентифікування котів в інших зображеннях. Вони роблять це без жодного апріорного знання про котів, наприклад, що вони мають хутро, хвости, вуса та котоподібні писки. Натомість, вони розвивають свій власний набір доречних характеристик з навчального матеріалу, який вони оброблюють.

Як альтернатива підгонці даних до рівняння та звіту про отримані з нього коефіцієнти, нейронні мережі призначені для обробки вхідної інформації та створення прихованих моделей взаємозв’язків. Однією з переваг нейронних мереж є те, що вони природно здатні моделювати нелінійні системи. Недоліки включають тенденцію переповнювати дані та значний рівень труднощів у визначенні того, які дескриптори є найбільш значущими в отриманій моделі. У останніх дослідженнях QSAR RBFNN (Radial Basis Function Neural Network) і

GRNN (General Regression Neural Network) є найбільш часто використовуваними серед NN.

RBFNN складається з трьох шарів: вхідного, прихованого та вихідного. Вхідний рівень не обробляє інформацію; він лише розподіляє вхідні вектори на прихований шар. Кожен нейрон на прихованому шарі використовує радіальну базисну функцію як нелінійну функцію передачі для роботи з вхідними даними (рис. 3.3.1). Загалом, існує кілька радіальних базисних функцій (RBF): лінійна, кубічна, тонка пластинчаста сплайн, гаусова, мультиквадратична та обернена мультиквадратична. Найбільш часто використовуваною RBF (Radial Basis Function) є функція Гаусса, яка характеризується центром і шириною.

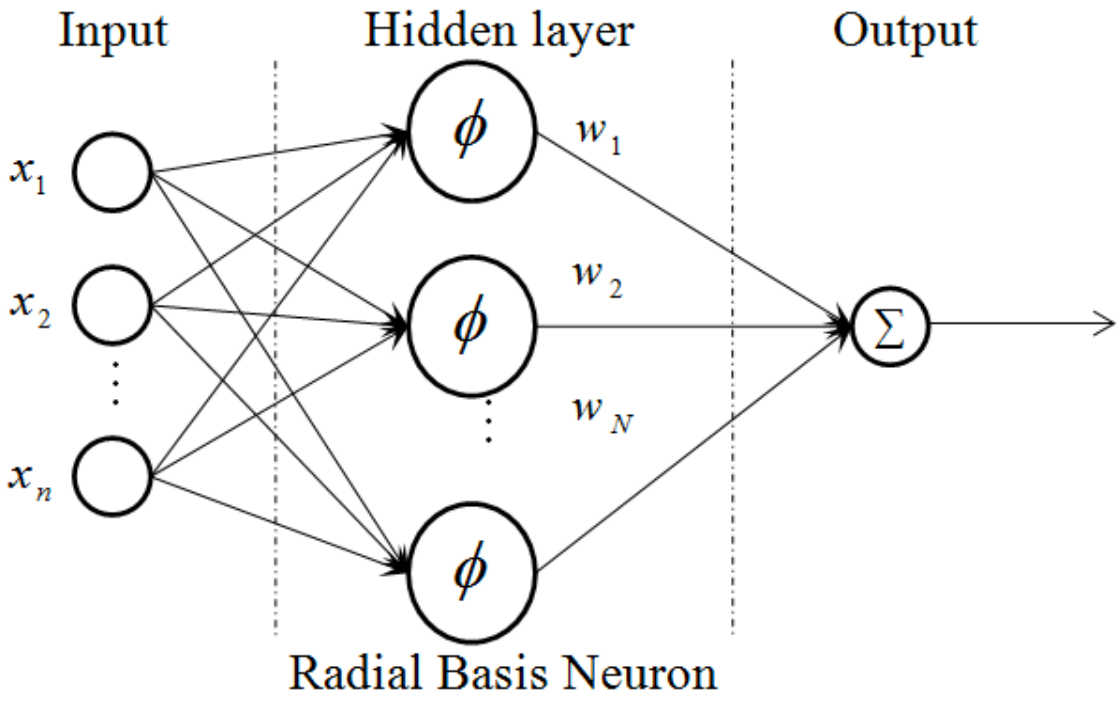

Рис. 4.3.1 Структура RBFNN

Загалом використання штучних нейронних мереж має багато переваг, таких як:

а) Швидкість роботи алгоритмів

б) Найбільша гнучкість та точність отриманих результатів

в) Можливість глибокого аналізу та навіть отримання власних дескрипторних просторів

Порівняння точності результатів нейронних мереж та лінійної регресії, яка використовується у ролі фітнес функції евристичних алгоритмів можна побачити на рисунку 4.3.2:

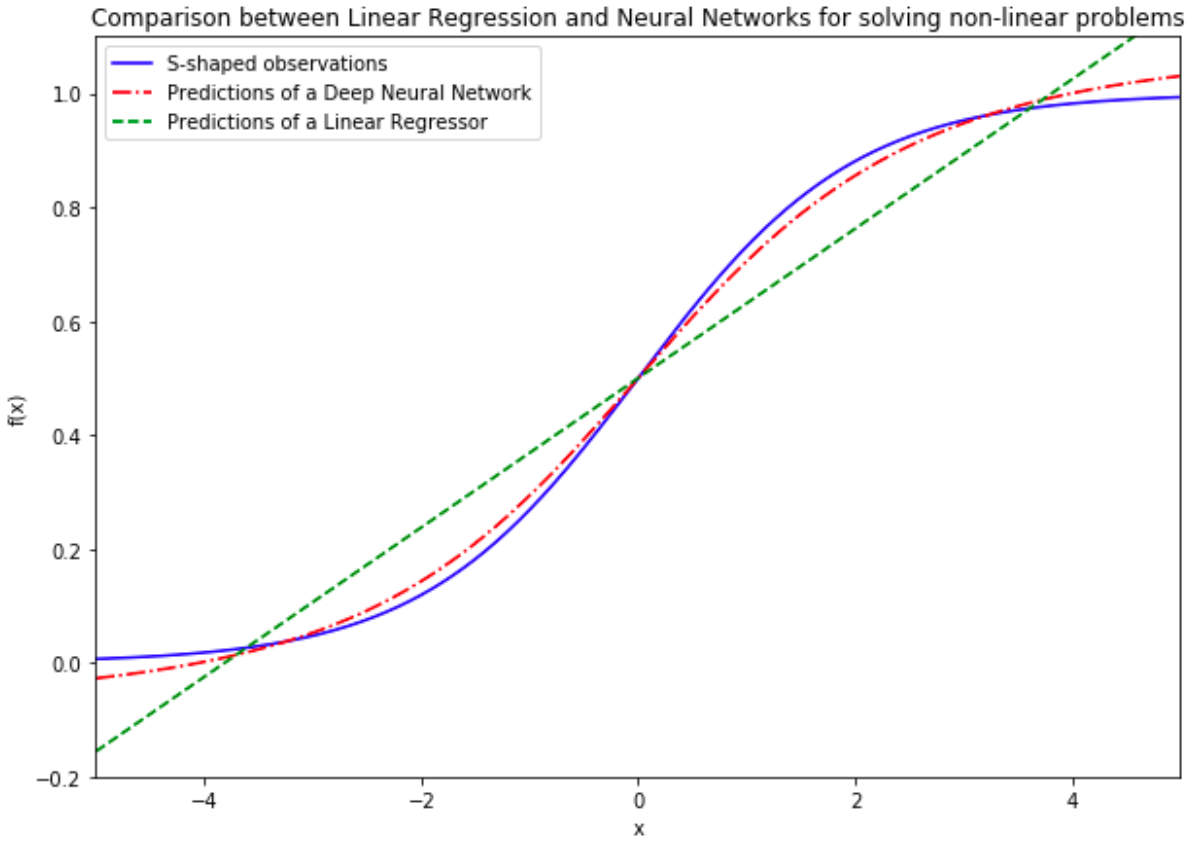

Рис. 4.3.2 Порівняння нейронних мереж та лінійної регресії

Та деякі недоліки також роблять цей підхід менш популярним при роботі з QSAR. Серед них:

а) Потреба у найбільшій кількості ресурсів

б) Необхідність у роботі с GPU під час навчання моделі

в) Проблема інтерпретації моделі

Саме остання проблема є найбільш важливою, бо порівняно з статистичними методами вирішення QSAR, NN не дають можливості побачити кореляцію між проміжними результатами дескрипторних залежностей та фінальною формулою отриманою у ході роботи алгоритму.

5. МЕТОД ОПТИМІЗАЦІЇ МУРАШИНИХ КОЛОНІЙ ДЛЯ ВИРІШЕННЯ ЗАДАЧ КОМП'ЮТЕРНОЇ ХІМІЇ

5.1Метод оптимізації мурашиних колоній для обирання ознак у QSAR

Кількісне співвідношення структура-активність (QSAR) здійснює пошук інформації, що стосується хімічної структури з біологічною та іншими видами діяльності, розробляючи модель QSAR. Стадії роботи QSAR можна поділити на декілька кроків (рис. 5.1.1):

а) Збір бази інформації сполук
б) Відбір необхідних дескрипторів
в) Вибір моделі для аналізу
г) Аналіз дескрипторів з метою пошуку невеликої підмножини, яка результат достатньої влучності
д) Валідація результатів аналізу

Алгоритм має давати найкращі значення за швидкістю виконання, якщо стадії схильні до високої ступені паралелізації.

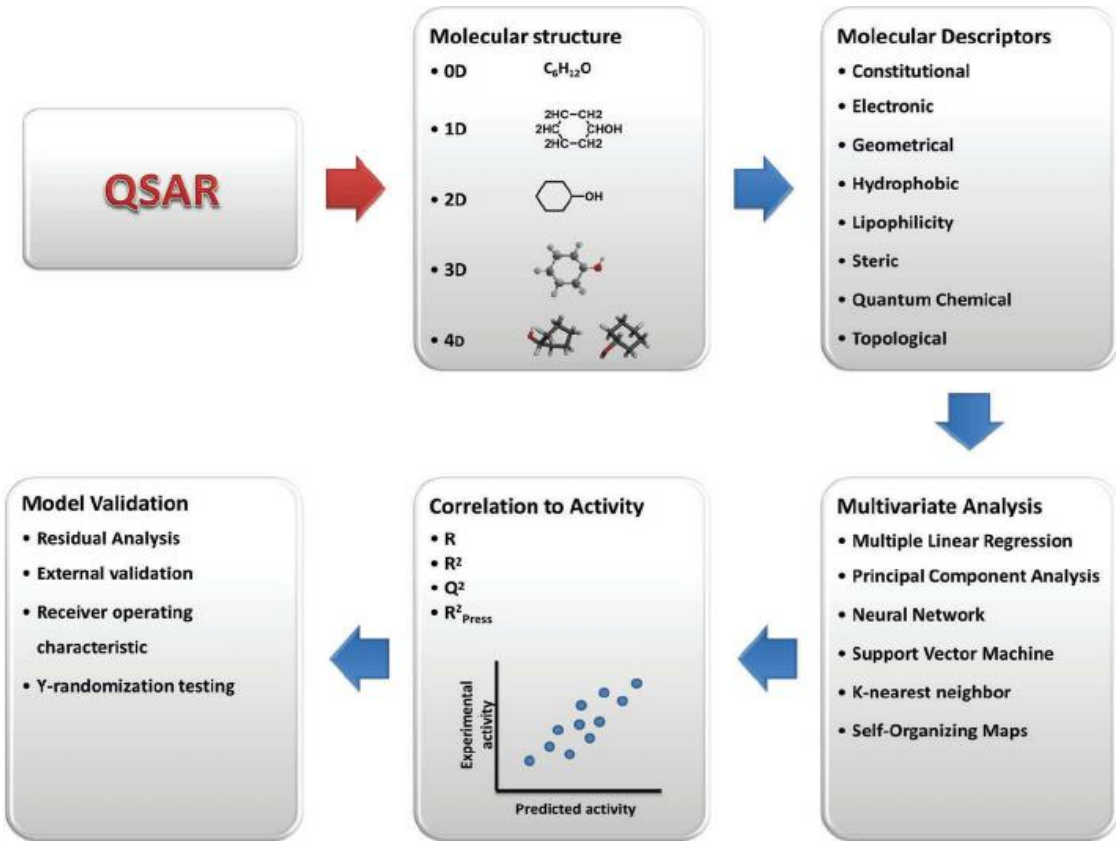

Рис. 5.1.1 Стадії QSAR

Експериментальна інформація може асоціюватися з біологічними властивостями, такими як активність, токсичність або біодоступність, які приймаються як залежні змінні при побудові моделі. Хімічна структура представлена різноманітними дескрипторами, включаючи просторові, електронні, топологічні, інформаційно-змістові, термодинамічні, конформаційні, квантово-механічні та дескриптори форми. Кілька сотень навіть тисяч дескрипторів можуть бути сформовані в дослідженнях QSAR. Але лише частина з них є статистично значущою з точки зору кореляції з біологічною активністю для конкретного аналізу, і вибір змінних необхідний для створення корисної прогностичної моделі. У QSAR кількість сполук із наявними значеннями біологічної активності зазвичай невелика порівняно з кількістю структурних дескрипторів. Це може призвести або до можливого переобладнання, або навіть до повного провалу в побудові значущої моделі регресії. Вибір змінних, які дійсно вказують на відповідну біологічну активність, стає одним із ключових кроків у дослідженнях QSAR. Вигода від

варіативного відбору в QSAR полягає не тільки в стабільності моделі, але і в інтерпретації взаємозв'язку між дескрипторами та біологічною активністю.

Вибір змінних у QSAR є проблемою вибору підмножини. Є декілька різних підходів до вибору підмножини дескрипторів (рис 5.1.2):

а) Фільтрові методи

б) Обгорткові методи

в) Вбудовані методи

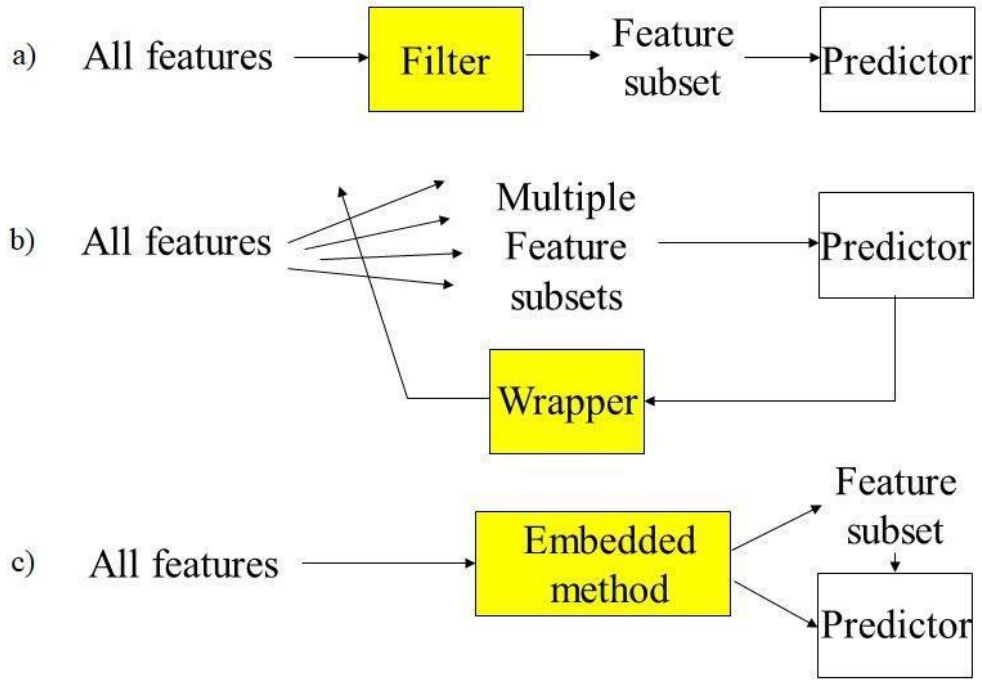

Рис. 5.1.2 Класифікація методів відбору ознак (feature selection)

Обгорткові методи для встановлення балів підмножинам ознак використовують передбачувальну модель. Кожну нову підмножину використовують для тренування моделі, яку перевіряють на притриманій множині. Підрахунок кількості помилок, зроблених на цій притриманій множині (рівень похибки моделі) дає бал цієї підмножини. Оскільки обгорткові методи тренують нову модель для кожної підмножини, вони є дуже обчислювально напруженими, але зазвичай пропонують множину ознак із найкращою продуктивністю для того окремого типу моделі.

Фільтрові методи для встановлення балів підмножинам ознак замість рівня похибки використовують міру-заступницю. Цю міру обирають такою, щоби вона була швидкою, але все ще схоплювала корисність множини ознак. До поширених мір належать взаємна інформація, поточкова взаємна інформація, коефіцієнт кореляції Пірсона, алгоритми на основі Relief та внутрішньо/міжкласова відстань або бали критеріїв значущості для кожної комбінації клас/ознака. Фільтри зазвичай є менш напруженими обчислювально за обгортки, але вони пропонують набір ознак, що не налаштовано на конкретний тип передбачувальної моделі. Цей брак налаштування означає, що набір ознак від фільтра є загальнішим за набір від обгортки, зазвичай даючи нижчу передбачувальну продуктивність, ніж обгортка. Проте такий набір ознак не містить припущень передбачувальної моделі, і тому є кориснішим для розкриття взаємозв'язків між ознаками. Багато фільтрів пропонують ранжування ознак замість явної найкращої підмножини ознак, а точку відсікання в рангу обирають за допомогою перехресного затверджування. Фільтрові методи також застосовували як передобробний етап для обгорткових методів, роблячи можливим застосування обгорток до більших задач. Одним з інших популярних підходів є алгоритм рекурсивного усунення ознак, зазвичай застосовуваний з методом опорних векторів для повторної побудови моделі та усунення ознак з низькими ваговими коефіцієнтами.

Вбудовані методи є всеосяжною групою методик, що виконують обирання ознак як частину процесу побудови моделі. Примірником цього підходу є метод побудови лінійних моделей LASSO, який штрафує коефіцієнти регресії штрафом L1, скорочуючи багато з них до нуля. Будь-які ознаки, що мають ненульові коефіцієнти регресії, є «обраними» алгоритмом LASSO. До покращень LASSO належать Bolasso, яке бутстрепує вибірки, еластично-сіткова регуляризація, яка поєднує штраф L1 LASSO із штрафом L2 гребеневої регресії, та FeaLect, яке встановлює бали всім ознакам на основі комбінаторного аналізу коефіцієнтів регресії. Ці підходи з погляду обчислювальної складності тяжіють до знаходження між фільтрами та обгортками.

У модифікованому ACO не було концепції шляху. Імовірність переміщення зі значенням 1 або 0 посилалася на кожен вимір, а не на кожен шлях. Обчислювали рівні феромонів, що відповідають розмірності, що приймає значення 1 або 0, а потім розраховували ймовірність переміщення відповідно до кількості феромонів, щоб визначити, чи було обрано змінну в наступній ітерації. У модифікованому ACO, чим більше залишається феромонного сліду τ_{i1} на змінній *i*, тим більш вірогідною буде вибрана ця змінна [8]. Механізм обміну інформацією однаковий у модифікованому ACO та звичайному ACO. У двох версіях ACO особа оновлюється відповідно до інформаційних позитивних відгуків та механізму непрямого спілкування. Використовуючи попередню найкращу інформацію про кожного мураха, модифікований ACO досить швидко сходить до оптимального положення із задовільними збіжними характеристиками. У модифікованому ACO рівні феромонів оновлювались не лише за допомогою інформації про поточного індивіда, але й за попередніми або загальносвітовими найкращими показниками кожного мураха, тому позитивні відгуки інформації в модифікованому ACO насправді відрізнялися від таких у звичайному ACO.

5.2. Конструювання пептидів за допомогою методу оптимізації мурашиних колоній

В даний час пептиди переживають епоху відродження як інструментальні та головні структури у фармацевтичних дослідженнях та хімічній біології, зокрема завдяки поєднанню обчислювального, хімічного та біологічного підходів. Хоча твердофазний синтез та тестування активності in vitro дозволяють аналізувати одночасно кілька тисяч пептидів, вичерпні пептидні бібліотеки стають надмірно неможливими із збільшенням довжини пептидів. Як елегантна альтернатива, технології відображення фагів пропонують паралельний доступ приблизно до 1015 послідовностей. Хоча цей біохімічний підхід забезпечить пептиди з бажаними

властивостями та діяльністю для широкого спектру застосувань, він також страждає від кількох обмежень. Наприклад, дуже гідрофобні послідовності та пептиди, які вбивають або іншим чином впливають на бактерії-господаря, що використовуються для виробництва фагів, уникнуть ідентифікації шляхом фагового дисплея. Коли час і ресурси обмежені, а гіпотеза моделі чи дизайну доступна, комп'ютерна генерація пептидів de novo є альтернативою для вирішення проблеми оптимізації комбінаторного пептиду. Ця методологія заснована на прогностичній моделі пептидної активності та надійному методі оптимізації для навігації в просторі послідовностей до областей високопрогнозованої придатності.

Для цільового відбору проб пептидів може бути використаний метод, який називається Молекулярний мурашиний алгоритм (MAntA), який реалізує алгоритм ACO. Це моделює процес того, як мурахи знаходять найкоротший шлях між джерелом їжі та своїм гніздом, використовуючи феромонові стежки [9]. MAntA переглядає цю концепцію таким чином, щоб колонія штучних пошукових агентів (мурах) будувала путі (пептиди), оцінювала якість путі (придатність, виміряна активність) та оновлювала феромонні сліди мурах пропорційно якості путі. Конструкція пептидів змодельована як проблема відбору підмножини з обмеженням, що для кожної позиції має бути обрана рівно одна амінокислота. Феромони безпосередньо осідають на вузлах, які відповідають амінокислотам у кожному положенні. На кожному етапі побудови пептиду мурахи вибирають наступний крок путі (амінокислоту) на основі правила імовірнісного рішення. Цей імовірнісний вибір упереджений шляхом феромонів та наявною евристичною інформацією. Стежка феромонів оновлюється після кожного циклу проектування відповідно до прогнозів за допомогою функції фітнесу (модель машинного навчання). Лише амінокислотні залишки найбільш придатного розчину (тобто найкращого в даний час пептиду з точки зору підрахованого балу) отримують додатковий феромон, що математично виражається збільшенням ймовірностей. Отже, у пошуковому просторі з'являються переважні мурашині шляхи, які відповідають потенційним високоякісним амінокислотним послідовностям. Щоб

зменшити ризик ранньої конвергенції процесу пошуку, невелика частка феромонів випаровується після кожного циклу.

Схематично алгоритм продемонстрований на рисунку 5.2.1

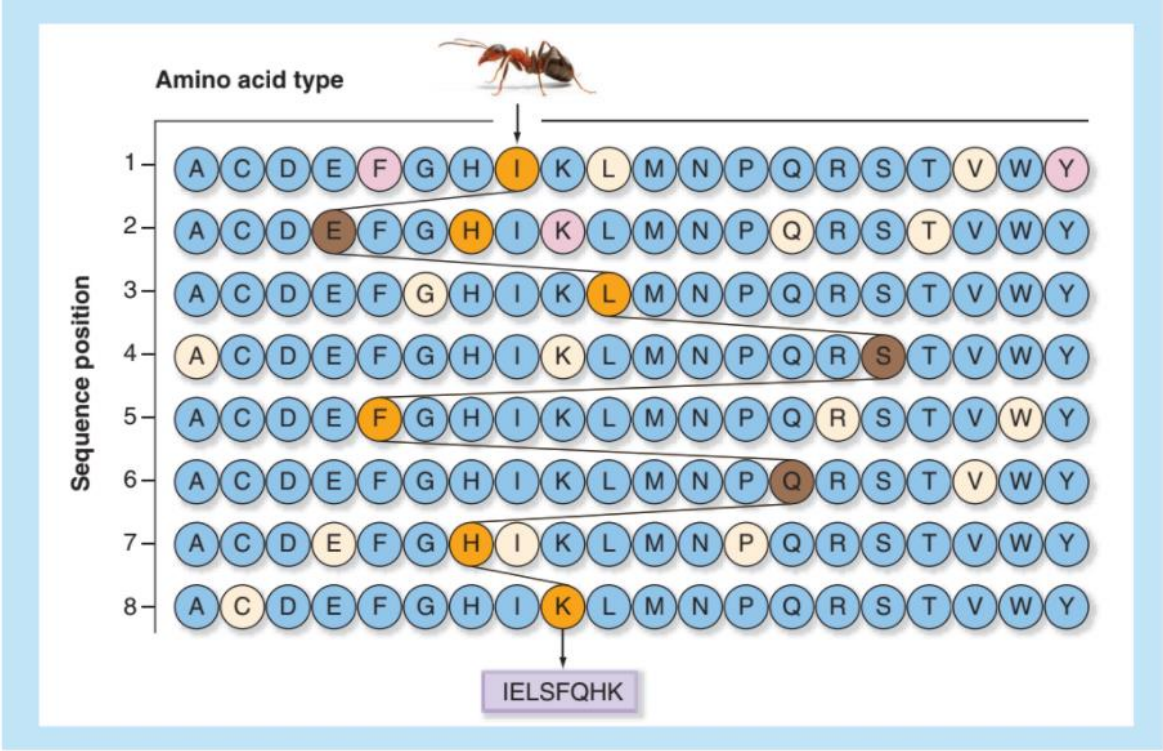

Рис. 5.2.1 Алгоритм MAntA

5.3. Молекулярний докінг протеїн-лігандів за допомогою методу оптимізації мурашиних колоній

Так звана проблема стикування білка-ліганду (protein-ligand docking problem – PLDP) вперше сформульований Фішером, використовуючи відому метафору «замок-ключ»: ключ (ліганд) повинен точно вписуватися в замок (білок), щоб відкрити двері (фармакологічний ефект). Алгоритм стикування намагається вирішити проблему прогнозування положення, яка полягає у пошуку правильної орієнтації та конформації ліганду в апріорно відомому активному центрі білка (рис 5.3.1).

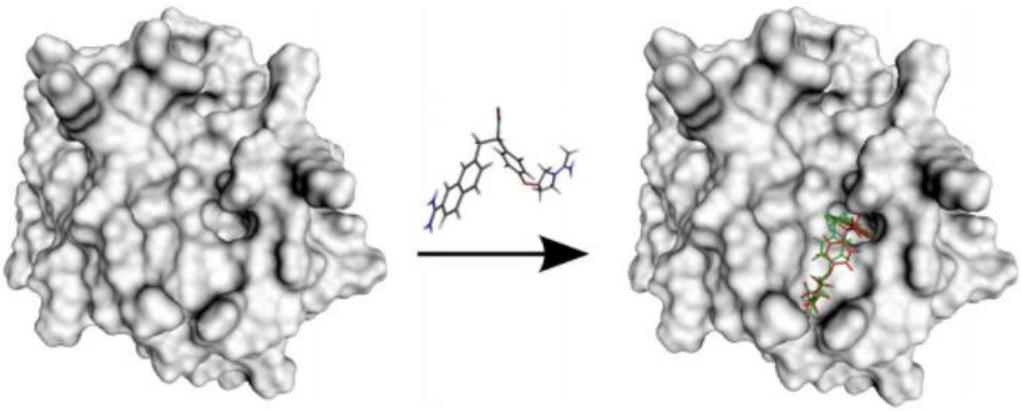

Рис. 5.3.1 Проблема передбачення положення елементу

Отримані комплекси оцінюються та сортуються відповідно до цільової функції (підрахункової функції), яка може бути інтерпретована як міра їх спорідненості до зв'язування. Потім найкращі бали лігандів можна перевірити експериментальними методами біологічної активності. Іноді немає структурної інформації про мішень, але інформація про активність відомих зв'язуючих лігандів не доступна. У цій ситуації можуть застосовуватися так звані методи на основі ліганду. Основний принцип цих підходів полягає у використанні подібності біологічно активних лігандів для пошуку подібних нових [9]. Методи в цій галузі варіюються від простих одновимірних фільтрів, таких як кількість атомів, молекулярна вага тощо, через пошук на основі підструктури до тривимірних пошуків за допомогою фармакофорних моделей. Як і у випадку, що базується на структурі, база даних лігандів перевіряється за допомогою цих методів і сортується відповідно до подібності або функції оцінки.

Простір, який шукає алгоритм PLANTS (Protein–Ligand ANT System), визначається поступальною, обертальною та крутильною ступенями свободи ліганда, а також крутильною ступенем свободи білка. Оскільки ACO спочатку розроблений для вирішення комбінаторних задач, дискретизуються неперервні змінні та застосується MAX –MIN Ant System (MMAS). Дискретизація використовує для кожного з трьох поступальних ступенів свободи інтервал довжиною 0,1 Å, тоді як для трьох ступенів свободи обертання та всіх ступенів

свободи кручення береться взятий інтервал довжиною 1°, що дає 360 значень для останнього. Кількість значень для кожного поступального ступеня свободи залежить від діаметра заздалегідь визначеного розміру ділянки зв'язування. Для багатьох комплексів, ця дискретизація призводить до 120-400 дискретних точок для кожного виміру *x*, *y* та *z*. Кожен ступінь свободи і асоціював феромонний вектор τ_i з такою кількістю записів, скільки значень, отриманих в результаті дискретизації. Потім феромоновий слід τ_{ij} стосується бажаності присвоєння значення *j* ступеню свободи *i*.

5.4Придатність алгоритмів QSAR для вирішення хімічних задач

Отже, виходячи з аналізу задач комп’ютерної хімії ми можемо сказати, що здатність алгоритмів ACO вирішувати вищеназвані завдання базується на деяких відмінностях:

а) Кількість ступенів свободи дескрипторного простору
б) Алгоритм верифікації обраного мурахами шляху
в) Паралелізація стадій алгоритму
г) Складність пошуку помилок та налагодження алгоритму

Спираючись на це ми можемо зробити наступні висновки:

а) Конструювання пептидів може бути аналітично виконано за допомогою алгоритмів типу ACO, але неможливість оцінювати хімічні якості білку до стадії in vitro ускладнюють пошук необхідного шляху та scoring моделі.
б) Молекулярний докінг протеїн-лігандів має дуже прогнозовану функцію оцінки результатів роботи моделі на кожному шляху, але має обмеження пов’язані з збільшеною кількістю ступенів свободи та комплексністю алгоритму, що ускладнює процес відлагодження алгоритму у 3-х вимірному просторі

в) Задача QSAR є найбільш придатною для використання ACO, бо має кроки, які легко паралелізуються. Алгоритм працює у 2-х вимірному просторі з відносно простою структурою графу, що полегшує пошук помилок та адаптацію моделі

6. РЕАЛІЗАЦІЯ ВИРІШЕННЯ ЗАДАЧІ QSAR ЗА ДОПОМОГОЮ МЕТОДУ ОПТИМІЗАЦІЇ МУРАШИНИХ КОЛОНІЙ

6.1. Опис роботи алгоритму

В даній роботі представляється програмний комплекс для вирішення задачі QSAR за допомогою модифікованого алгоритму оптимізації мурашиних колоній.

У мурах ACO використовується хімічна речовина, яка називається феромоном, для зв’язку між собою, і цей процес характеризується позитивним циклом зворотного зв’язку: чим більше мурах використовує певний шлях, тим більше феромону осідає на цьому шляху, і тим більше він стає привабливий для інших мурах. Комунікація та позитивні відгуки – основний механізм алгоритму ACO. Випаровування феромонів необхідне, щоб уникнути занадто швидкого зближення алгоритму до неоптимальної області і виступає за дослідження нових областей простору пошуку. Покращений механізм позитивного зворотного зв'язку, що означає, що велика кількість феромону, що утримується, може не тільки пришвидшити швидкість конвергенції, але й зробити можливим зближення алгоритму до локальних оптимумів. Проблеми вибору підмножини досить сильно відрізняються від задач упорядкування. Вибір підмножини означає вибір найкращого підмножини з цілого набору. Тут немає поняття шляху, тому важко застосувати звичайний ACO безпосередньо до вибору змінних у QSAR. Відповідно до інформаційних позитивних відгуків та механізму непрямого спілкування ACO пропонується модифікований ACO для вибору змінних. Для проблеми вибору змінної, вираженої у двійковому записі, мураха рухається в N-мірному просторі пошуку з N змінних, його рух обмежений 0 або 1 для кожного виміру. Стан "1" представляє вибір цієї змінної, а стан "0" – зворотний. У проблемі відбору бінарних змінних кожен мураха вибирає змінні, які визначались за імовірністю переміщення 0 або 1. Рівні феромонів у кожному вимірі (змінній), а не на шляху, поділяються на два типи, τ_{i0} та τ_{i1}, які представляють феромон розмірності і, що приймає значення

1 та 0 відповідно. Рівні феромонів, що відповідають розміру, що приймає значення 1 або 0, оновлюються відповідно до правила оновлення.

$$\tau_{i0}(new) = \rho\tau_{i0}(old) + \Delta\tau_{i0} \qquad (1)$$

$$\Delta\tau_{i0} = \sum_{k=1}^{m} \Delta\tau_{i0}^{(k)} \qquad (2)$$

$$\tau_{i1}(new) = \rho\tau_{i1}(old) + \Delta\tau_{i1} \qquad (3)$$

$$\Delta\tau_{i1} = \sum_{k=1}^{m} \Delta\tau_{i1}^{(k)} \qquad (4)$$

де $\Delta\tau_{i0}$ та $\Delta\tau_{i1}$ представляли приріст феромону, що відповідає розмірності *i*, що приймає значення 1 або 0 у цьому колі. $\Delta\tau_{i0}$ (*k*) та $\Delta\tau_{i1}$ (*k*) показують кількість феромону, який мураха *k* залишила на змінній і у цьому колі. Для кожного виміру інтенсивність феромону в момент часу 0 (τ_{i0} і τ_{i1}) встановлюється рівною 0.

Ant k приймає рішення щодо варіативного відбору відповідно до кількості феромонів. Імовірність переміщення становить:

$$p_i^{(k)} = \frac{\tau_{i1}}{\tau_{i1} + \tau_{i0}} \qquad (5)$$

У модифікованому ACO мурахи вибирають змінні з усіх *N* змінних відповідно до ймовірності, визначеної рівнянням 5. Після одного відбору кількість феромону оновлюється відповідно до рівнянь 1-5. Цей процес повторюється доти, доки не буде досягнуто критерій мінімальної помилки або кількість ітерацій не досягне визначеного користувачем обмеження.

6.2. Реалізація алгоритму

Для реалізації модифікованого алгоритму оптимізації мурашиної колонії була обрана мова програмування Python з бібліотеками Sklearn, Pandas та NumPy (додаток А). Тестування результатів алгоритму було проведено на датасеті з 373 органічних ароматичних сполук з базою дескрипторів, яка складалась з 1497 автоматично згенерованих дескрипторів.

У якості фітнес функції алгоритму було обрано множинну лінійну регресію (MLR), значення кожної підмножини дескрипторів були розбиті на train та test набори у відношенні 70% та 30%. У якості оцінки моделі було взято значення R^2 .

Для тестування якості роботи моделі були обрані табличні значення (табл. 6.2.1) критичних: температури, тиску, та обсягу.

Табл. 6.2.1 Перші 30 сполук та їх значень критичних температури, тиску, та обсягу

number	name	critical temperature	critical pressure	critical volume
1	acetophenone	701	3840000	0,376
2	1-chloro-2-nitro-4-(trifluoromethyl)benzene	686	2740000	0,49
3	3-chloroaniline	751	4590000	0,364
4	chlorobenzene	632,35	4519000	0,308
5	1-chloro-4-(trifluoromethyl)benzene	601	3010000	0,399
6	3-chlorobenzoyl chloride	724	3680000	0,406
7	1-chloronaphthalene	785	3400000	0,434
8	2,4-dimethylphenol	707,65	4400000	0,39
9	1-chloro-4-methylbenzene	660	3910000	0,36
10	1-chloro-3-methylbenzene	660,18	3853000	0,3685
11	m-cresol	705,85	4560000	0,312
12	1-hydroperoxy-4-isopropylbenzene	605	3340000	0,419
13	(oxybis(methylene))dibenzene	777	2560000	0,608
14	1,3-dibromobenzene	761	4660000	0,372
15	dibutyl phthalate	781	1750000	0,846
16	1,3-dichlorobenzene	683,95	4070000	0,351
17	2,4-dichloro-1-(trifluoromethyl)benzene	646	2810000	0,443
18	2,4-dichloro-1-methylbenzene	705	3590000	0,404
19	2,6-diethylaniline	678	3120000	0,495
20	diethyl phthalate	757	2330000	0,635
21	1,3-difluorobenzene	552,94	4067000	0,2995
22	1,4-difluorobenzene	556	4400000	0,2995
23	1-methoxy-4-(prop-1-en-1-yl)benzene	723	2900000	0,482
24	aniline	699	5309000	0,27
25	anisole	641,65	4175000	0,337
26	benzaldehyde	695	4650000	0,324
27	benzonitrile	699,35	4215000	0,339
28	(trichloromethyl)benzene	737	3340000	0,447
29	(trifluoromethyl)benzene	565	3390000	0,356
30	benzoyl chloride	697	4060000	0,367

6.3. Результати алгоритму

Алгоритм ставив мету пошуку найліпшої моделі, яка описує залежність у 2, 3 та 4-дескрипторному просторі. Для перевірки результатів те ж саме завдання ставилось для алгоритмів повного перебору та генетичному алгоритму. Алгоритм повного перебору був протестований тільки на 2 та 3-дескрипторному просторі

через велику обчислювальну складність повного перебору 4-дескрипторного простору.

Результати генетичного та алгоритму оптимізації мурашиних колоній збігались на більшості ітерацій та відповідають результатам повного перебору (табл 6.3.1). Варто зазначити, що генетичний та алгоритм оптимізації мурашиної колонії на 4-дескрипторному просторі показують досить схожу швидкість роботи у межах 2-3 хвилин.

Табл 6.3.1 Результати роботи алгоритму

Property name	2 params - R^2	3 params - R^2	4 params - R^2
Critical temperature	[‘MATS1v’, ‘X5sol’] - 0.7050	[‘MATS1v’, ‘Ui’, ‘X3sol’] - 0.7498	[‘MATS1v’, ‘Ui’, ‘X3sol’, ‘nF’] - 0.7825
Critical pressure	[‘HIC’, ‘nOHPh’] - 0.6153	[‘C-001’, ‘SP03’, ‘nOHPh’] - 0.7343	[‘C-001’, ‘SP02’, ‘nOHPh’, ‘nP’] - 0.8268
Critical volume	[‘Sv’, ‘nOHPh’] - 0.9205	[‘CIC0’, ‘Sv’, ‘nOHPh’] - 0.9468	[‘Sv’, ‘XMOD’, ‘nOHPh’, ‘nP’] - 0.9530

Також було проведено інтеграцію алгоритму до існуючої web системи «Chemistry Assembler» (табл. 6.3.2).

Табл. 6.3.2 Тест кейс інтеграційного тестування підключення алгоритму до системи «Chemisty Assembler»

Назва	Тест пошуку за QSAR	
Тип	Інтеграційний тест	
Дія	Очікуємий результат	Результат тесту
Передумова:		
Обрати базу даних для пошуку на сторінці вибору	База обрана, сторінка пошуку не показує, що база відсутня	Passed
Кроки:		
Обрати параметр кількості мурах	Форма приймає числове значення	Passed
Обрати параметр кількості ітерацій	Форма приймає числове значення	Passed
Обрана назва колонки, яка вважається шуканою	Сервер проводить валідацію значення, форма приймає результат	Passed
Кнопка «Search» натиснута	Знайдена формула у форматі LaTeX з'являється на екрані	Passed

6.4. Аналіз результатів алгоритму

Серед сильних сторон алгоритму можна зазначити:

а) Дуже висока здатність до паралелізації обчислень алгоритму. Кожна мураха може бути обчислена різними вузлами кластеру, тому алгоритм дуже гарно підходить до парадигми MapReduce, де операцією Map є пошук путі окремою мурахою, а операцією Reduce є агрегація путів та виділення феромону.

б) Висока гнучкість алгоритму. Алгоритм може у якості моделі використовувати досить багатий спектр підходів, від множинної лінійної регресії до нейронних мереж.

в) Алгоритм дає стабільно високі результати за дискретний час.

Незважаючи на вищеназване, алгоритм також має і декілька недоліків:

а) Алгоритм дуже схильний до локальних максимумів. На деяких ітераціях, якщо спочатку було обрано влучний, але не найоптимальніший дескриптор, то алгоритм буде заохочувати його вибір у наступних ітераціях підмножин

б) Алгоритм дуже залежний від евристичних параметрів. Алгоритм має багато модифікацій, які нівелюють ті, чи інші параметри за допомогою стохастичних параметрів. На різних датасетах може мати місце необхідність адаптивної зміни значень таких параметрів.

6.5. Перспективи покращення алгоритму

Незважаючи на те, що за допомогою алгоритму отримано непогані результати застосування як з точки зору швидкості виконання так і точності використання, є декілька аспектів, які можуть бути покращені, а саме:

а) Фітнес модель

б) Дескрипторний простір

в) Ступінь паралелізації

В якості фітнес моделі була обрана множинна лінійна регресія, що є однією з найпростіших моделей, але дає дуже передбачуваний результат. Розраховується, що використання SVM (Support-Vector machine), або нейронних мереж може давати кращий результат, але знижуючи при цьому здатність аналізувати модель.

З точки зору дескрипторного простору, алгоритм працював з статично згенерованими дескрипторами. Якщо використовувати в якості моделі нейронні мережі, то дескриптори можуть бути отримані у ході роботи алгоритму.

Ступінь паралелізації також може бути збільшена при використанні таких програмних комплексів як Spark або Hadoop та оптимізації за допомогою shuffling технологій, які вбудовані в ці системи.

ВИСНОВКИ

У ході кваліфікаційної роботи було проведено дослідження задачі QSAR, методу оптимізації мурашиних колоній, задач комп'ютерної хімія для яких є перспектива використання методу оптимізації мурашиних колоній. Таким чином було знайдено 3 сфери використання методу оптимізації мурашиних колоній:

а) Задача QSAR
б) Конструювання пептидів
в) Молекулярний докінг протеїн-лігандів

Також було створено авторський алгоритм для вирішення задачі QSAR та проведено його експериментальну оцінку. Для порівняння було обрано генетичний алгоритм та повний перебор дескрипторного простору. На основі досліджень можна зробити такі висновки:

а) Алгоритм оптимізації мурашиних колоній має дуже гнучку теоретичну базу та багато модифікацій, які роблять його універсальним у застосування для часткового рішення *NP*-повних задач.
б) Алгоритм оптимізації мурашиних колоній має перспективи бути використаним для рішення задач QSAR, конструювання пептидів, молекулярного докінгу протеїн-лігандів, а також інших проблем молекулярного моделювання
в) Алгоритм оптимізації мурашиних колоній є життєздатною альтернативою генетичному алгоритму, маючи схожі плюси та мінуси

Отримане у ході експериментів рішення можна використовувати як основу для подальшого застосування для вирішення задачі QSAR та подальших покращень та оптимізацій.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Leshchynskyi V. Modeling the user's choice in the constraints of the cold start of the recommender system / V. Leshchynskyi, I. Leshchynska. // Bionics of Intelligence. – 2019. – №2663.

2. QSAR – Вікіпедія: веб-сайт. URL: https://uk.wikipedia.org/wiki/QSAR (дата звернення: 23.01.2022).

3. Dorigo, M., St¨utzle, T.: Ant Colony Optimization. MIT Press, Cambridge, MA, USA (2004)

4. Kellenberger, E., Rodrigo, J., Muller, P., Rognan, D.: Comparative evaluation of eight docking tools for docking and virtual screening accuracy. Proteins 57(2) (2004) 225–242

5. Ant Colony Optimization: an introduction: веб-сайт. URL: https://ctlab.itmo.ru/~chivdan/presentations/aco-03-04-2013.pdf (дата звернення: 17.11.2021).

6. Cho, S. J.; Hermsmeier, M. A. Genetic Algorithm Guided Selection: Variable Selection and Subset Selection. J. Chem. Inf. Comput. Sci. 2002, 42, 927-936.

7. Shen, M.; LeTiran, A.; Xiao, Y.; et al. Quantitative Structure-Activity Relationship Analysis of Functionalized Amino Acid Anticonvulsant Agents Using k Nearest Neighbor and Simulated Annealing PLS Methods. J. Med. Chem. 2002, 45, 2811-2823.

8. Dorigo, M.; Gambardella, L. M. Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans. EVolutionary Comput. 1997, 1, 53-66.

9. Xiong, W. Q.; Wei, P. A kind of ant colony algorithm for function optimization. Machine Learning and Cybernetics. International Conference on Proceedings; 2002; Vol. 1, pp 552-555.