1. ПОСТАНОВКА ЗАДАЧІ

2.1 **Односторінковий опис проекту**

Розробка програмного забезпечення для розв’язання задач багатокритеріальної оптимізації генетичними алгоритмами. Короткий огляд. Розроблене програмне забезпечення повинне реалізовувати багатокритеріальну оптимізацію за допомогою генетичних алгоритмів багатокритеріальної оптимізації.

Вступ. Назва проекту: «Розробка програмного забезпечення для розв’язання задач багатокритеріальної оптимізації генетичними алгоритмами». Дата підготовки документа: 26 травня 2015р.

Описання особливостей поставки. Документ «Розробка програмного забезпечення для розв’язання задач багатокритеріальної оптимізації генетичними алгоритмами» буде доступний на CD-диску та в паперовій копії. Всі особливості специфікації будуть зазначені на CD-диску.

Користувач документа. Користувач – досвідчений програміст у галузі створення додатків з використанням мов HTML та Javascript, який розуміє переваги генетичних алгоритмів у вирішенні задач багатокритеріальної оптимізації.

Порівняльний аналіз. Генетичні алгоритми є передовими сучасними методами для вирішення задач багатокритеріальної оптимізації.

Описання технічного процесу: обрати відповідні генетичні алгоритми для багатокритеріальної оптимізації; визначити особливості програмної реалізації додатку, що виконує поставлені задачі; розробити проект керівництва користувача і визначити спосіб його розповсюдження; проаналізувати деталі реалізації, підготувати та включити в керівництво користувача відповідні приклади програм.

Залежності. З’ясувати залежність обраними початковими даними генетичних алгоритмів та результатами багатокритеріальної оптимізації. З’ясувати ефективність багатокритеріальної оптимізації залежно від використаного генетичного алгоритму.

Список основних документів. Специфікація розробленого додатку.

Основні дати. Завершення проектування «MultiobjectiveGA» – травень 2015р. Перший варіант повного «MultiobjectiveGA» – травень 2015р.

Ресурси. Ресурси які необхідні для реалізації даного проекту подані в таблиці 2.1.

Таблиця 2.1 – Ресурси для реалізації проекту

|  |  |  |
| --- | --- | --- |
| Функція | Ставка | Коментарі |
| Інженер-розробник | 0.75 | Консультуючий інженер, який реалізує розробку додатку |
| Технічний письменник | 0.25 | Підготовка документа |
| Керівник проекту | 0.3 | Керівництво, розробка плану і змісту |

2.2 **Постановка задачі багатокритеріальної оптимізації за допомогою генетичних алгоритмів**

2.2.1 **Загальний генетичній алгоритм**

Генетичні алгоритми виникли в результаті спостереження і спроб копіювання природних процесів, що відбуваються в світі живих організмів, зокрема, еволюції та пов'язаної з нею селекції (природного відбору) популяцій живих істот [22].

Ідею генетичних алгоритмів висловив Дж. Холланд у кінці шістдесятих — початку сімдесятих років XX століття. Він зацікавився властивостями процесів природної еволюції (в тому числі фактом, що еволюціонують хромосоми, а не самі живі істоти). Холланд був упевнений у можливості скласти і реалізувати у вигляді комп'ютерної програми алгоритм, який буде вирішувати складні задачі так, як це робить природа — шляхом еволюції.

Тому він почав працювати над алгоритмами, що оперували послідовностями двійкових цифр (одиниць і нулів), що одержали назву хромосом. Ці алгоритми імітували еволюційні процеси в поколіннях таких хромосом. У них були реалізовані механізми селекції та репродукції, аналогічно вживаними при природній еволюції [23 - 24].

Так само, як і в природі, **генетичні алгоритми**здійснювали пошук «добрих» хромосом без використання будь-якої інформації про характер розв'язуваної задачі. Була потрібна тільки якась оцінка кожної хромосоми, яка відображає її пристосованість. Механізм селекції полягає у виборі хромосом з найвищою оцінкою (тобто найбільш пристосованих), які репродукують частіше, ніж особини з більш низькою оцінкою (гірше пристосовані).

Репродукція означає створення нових хромосом у результаті рекомбінації генів батьківських хромосом. Рекомбінація — це процес, в результаті якого виникають нові комбінації генів. Для цього використовуються дві операції: схрещування, що дозволяє створити дві зовсім нові хромосоми нащадків шляхом комбінування генетичного матеріалу пари батьків, а також мутація, яка може викликати зміни в окремих хромосомах.

У генетичних алгоритмах застосовується ряд термінів, запозичених з генетики, перш за все гени і хромосоми, а також популяція, особина, алель, генотип, фенотип.

**Генетичні алгоритми** застосовуються при розробці програмного забезпечення, в системах штучного інтелекту, оптимізації, штучних нейронних мережах і в інших галузях знань. Слід зазначити, що з їх допомогою вирішуються завдання, для яких раніше використовувалися тільки нейронні мережі. У цьому випадку генетичні алгоритми виступають просто в ролі незалежного від нейронних мереж альтернативного методу, призначеного для вирішення тієї ж самої задачі [25].

**Генетичний алгоритм** являє собою метод, що відображає природну еволюцію методів вирішення проблем, і в першу чергу задач оптимізації. Генетичні алгоритми — це процедури пошуку, засновані на механізмах природного відбору і спадкоємства. У них використовується еволюційний принцип виживання найбільш пристосованих особин. Вони відрізняються від традиційних методів оптимізації декількома базовими елементами. Зокрема, генетичні алгоритми:

* обробляють не значення параметрів самого завдання, а їх закодовану форму;
* здійснюють пошук рішення виходячи не з єдиної точки, а з їх деякої популяції;
* використовують тільки цільову функцію, а не її похідні або іншу додаткову інформацію;
* застосовують імовірнісні, а не детерміновані правила вибору.

Перераховані чотири властивості, які можна сформулювати також як кодування параметрів, операції на популяціях, використання мінімуму інформації про завдання і рандомізація операцій приводять у результаті до стійкості генетичних алгоритмів і до їх переваги над іншими широко вживаними технологіями [1, 26-28].

При описі генетичних алгоритмів використовуються визначення, запозичені з генетики. Наприклад, мова йде про популяцію особин, а в якості базових понять застосовуються ген, хромосома, генотип, фенотип, алель. Також використовуються відповідні цим термінам визначення з технічного лексикону, зокрема, ланцюг, двійкова послідовність, структура.

**Популяція** — це кінцева множина особин.

**Особини**, що входять в популяцію, у генетичних алгоритмах представляються хромосомами з закодованими в них множинами параметрів задачі, тобто рішень, які інакше називаються точками в просторі пошуку (search points). У деяких роботах особини називаються організмами.

**Хромосоми** (інші назви — ланцюжки або кодові послідовності) — це впорядковані послідовності генів.

**Ген** (який також називається властивістю, знаком чи детектором) — це атомарний елемент генотипу, зокрема, хромосоми.

**Генотип** або структура — це набір хромосом даної особини. Отже, особинами популяції можуть бути генотипи або одиничні хромосоми (в досить поширеному випадку, коли генотип складається з однієї хромосоми).

**Фенотип** — це набір значень, які відповідає даному генотипу, тобто декодована структура або безліч параметрів задачі (розв’язок, точка простору пошуку).

**Алель** — це значення конкретного гена, також визначається як значення властивості або варіант властивості.

**Локус** чи позиція вказує місце розміщення даного гена в хромосомі (ланцюжку).

Дуже важливим поняттям у генетичних алгоритмах вважається **функція пристосованості** (fitness function), яка інакше називається функцією оцінки. Вона являє міру пристосованості даної особини в популяції. Ця функція відіграє найважливішу роль, оскільки дозволяє оцінити ступінь пристосованості конкретних особин у популяції і вибрати з них найбільш пристосовані (тобто мають найбільші значення функції пристосованості) відповідно з еволюційним принципом виживання «найсильніших» (які найкраще пристосувалися).

Функція пристосованості також отримала свою назву безпосередньо із генетики. Вона надає сильний вплив на функціонування генетичних алгоритмів і повинна мати точне і коректне визначення. У задачах оптимізації функція пристосованості, як правило, оптимізується і називається цільовою функцією.

У теорії управління функція пристосованості може приймати вигляд функції похибки, а в теорії ігор — вартісної функції [29].

На кожній ітерації генетичного алгоритму пристосованість кожної особини даної популяції оцінюється за допомогою функції пристосованості, і на цій основі створюється наступна популяція особин, що складають безліч потенційних рішень проблеми, наприклад, задачі оптимізації. Чергова популяція в генетичному алгоритмі називається поколінням, а до новостворюваної популяції особин застосовується термін «нове покоління» або «покоління нащадків».

Основний (класичний) генетичний алгоритм, що був використаний у розробленому програмному забезпеченні, складається з наступних кроків:

1. ініціалізація, або вибір вихідної популяції хромосом;
2. оцінка пристосованості хромосом в популяції;
3. перевірка умови зупинки алгоритму;
4. селекція хромосом;
5. застосування генетичних операторів;
6. формування нової популяції;
7. вибір «найкращої» хромосоми.

Блок — схема основного генетичного алгоритму зображена на рис. 2.1.

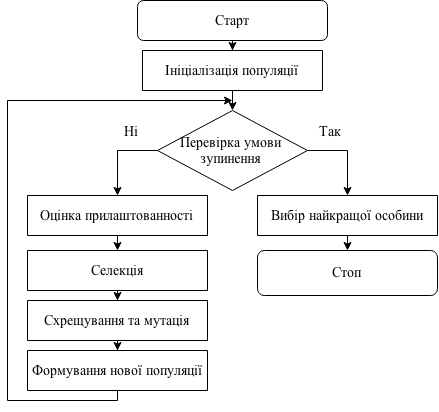
****

Рисунок 2.1 – Блок-схема генетичного алгоритму

Розглянемо конкретні етапи роботи додатку більш докладно.

Ініціалізація, тобто формування вихідної популяції, полягає у випадковому виборі заданої кількості особин.

Оцінювання пристосованості хромосом в популяції полягає в розрахунку функції пристосованості для кожної хромосоми цієї популяції. Для задачі мінімізації, чим менше значення цієї функції, тим вище «якість» хромосоми [1-4].

Визначення умови зупинки генетичного алгоритму залежить від його конкретного застосування. У оптимізаційних задачах, якщо відомо максимальне (або мінімальне) значення функції пристосованості, то зупинка алгоритму може відбутися після досягнення очікуваного оптимального значення, можливо — з заданою точністю.

Зупинка алгоритму також може статися у разі, коли його виконання не приводить до поліпшення вже досягнутого значення. Алгоритм може бути зупинений після закінчення певного часу виконання або після виконання заданої кількості ітерацій. Якщо умова зупинки виконана, то проводиться перехід до завершального етапу вибору «найкращої» хромосоми. В іншому випадку на наступному кроці виконується селекція [7].

Селекція полягає в тому, що батьками можуть стати тільки ті особини, значення пристосованості яких не менше граничної величини, наприклад середнього значення пристосованості по популяції. Такий підхід забезпечує швидшу збіжність алгоритму. Проте, через швидку збіжність селективний вибір батьківської пари не підходить тоді, коли ставиться завдання визначення декількох екстремумів, оскільки для таких завдань алгоритм, як правило, швидко сходиться до одного з рішень. Крім того, для деяких багатовимірних завдань зі складним ландшафтом цільової функції швидка збіжність може мати невірний розв'язок. Цей недолік може бути частково компенсований використанням відповідного механізму відбору, який би «гальмував» занадто швидку збіжність алгоритму. Гранична величина в селекції може бути обчислена різними способами. Тому в літературі для генетичного алгоритму виділяють різні варіації селекції. Найбільш відомі з них — це турнірний і рулеточний (пропорційний) відбори.

Рулеточний відбір не може бути застосований у задачах мінімізації функції, тому у додатку використовується турнірний відбір. Також турнірний відбір забезпечує більш точний результат, та більшу швидкість роботи.

При турнірному відборі (tournament selection) з популяції, яка складається із *N* особин, вибираються випадковим чином *t* особин, і найкраща особина записується в проміжний масив. Ця операція повторюється *N* раз. Особини в отриманому проміжному масиві потім використовуються для схрещування (також випадковим чином). Розмір групи рядків, що відбираються для турніру, часто дорівнює 2. У цьому випадку говорять про двійковий (парний) турнір. Взагалі ж *t* називають чисельністю турніру. Перевагою даного способу є те, що він не вимагає додаткових обчислень.

Одноточковий кросинговер (Single-point crossover) моделюється наступним чином. Нехай є дві батьківські особини з хромосомами і  . Випадковим чином визначається точка всередині хромосоми (точка розриву), в якій обидві хромосоми діляться на дві частини і обмінюються ними. Такий тип кросинговеру називаються одноточковим, так як при ньому батьківські хромосоми розділяються тільки в одній випадкової точці. Принцип роботи одноточкового кросинговеру зображений на рисунку 2.2.

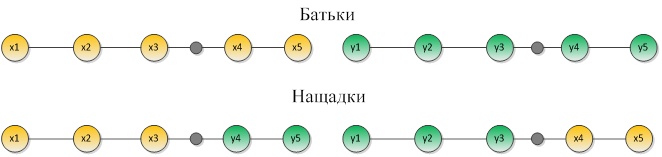


Рисунок 2.2 – Одноточковий кросинговер

Хромосоми, отримані в результаті застосування генетичних операторів до хромосом тимчасової батьківської популяції, включаються до складу нової популяції. Вона стає так званою поточною популяцією для даної ітерації генетичного алгоритму.

На кожній черговій ітерації розраховуються значення функції пристосованості для всіх хромосом цієї популяції, після чого перевіряється умова зупинки алгоритму і або фіксується результат у вигляді хромосоми з найбільшим значенням функції пристосованості, або здійснюється перехід до наступного кроку генетичного алгоритму, тобто до селекції. У класичному генетичному алгоритмі вся попередня популяція хромосом заміщається новою популяцією нащадків, що має ту ж чисельність.

Вибір «найкращої» хромосоми. Якщо умова зупинки алгоритму виконана, то слід вивести результат роботи, тобто представити шуканий розв'язок задачі. Кращим рішенням вважається хромосома з найбільшим значенням функції пристосованості.

2.2.2 **Метод RWGA (RANDOM Weights Genetic Algorithm)**

Цей підхід однин з найпопулярніших та є розвитком класичних методів бакатокритеріальної оптимізації, де нова загальна цільова функція будується з окремих цільових функцій у вигляді зваженої суми [30, 31].

Такий підхід був уперше використаний для отримання змінного напрямку пошуку фронту Парето. При фіксованих вагах в даному підході генетичний алгоритм відображає тенденцію постійного напряму пошуку, в той час як використання випадкових ваг відображає тенденцію змінного напрямку пошуку, більш пристосованою для пошуку фронту рішень.

Крок 1. Покласти =1 и проміжну популяцію =0.

Крок 2. Для кожної цільової функції встановити випадкову вагу

,

де *rj* - випадкові числа з відрізка [0,1];

*q* - число цільових функцій.

Крок 3. Вирахувати загальну фітнес функцію

,

де *wi* - значення ваги;

*fi(x)* - цільові функції.

Крок 4. За допомогою турнірного відбору обрати індивіда , ґрунтуючись на значенні придатності за даним критерієм, і скопіювати його у проміжну популяцію

:=+.

Крок 5. Провести для проміжної популяції операції схрещування та мутації, замінити батьківську популяцію на проміжну популяцію нащадків.

Крок 6. Покласти

=+1.

Якщо , перейти на крок 2, інакше стоп.

Скалярне значення фітнес-функції обчислюється шляхом підсумовування зважених значень цільових функцій. Для паралельного пошуку кратних рішень ваги не фіксуються, що дає можливість ГА шукати весь фронт в усіх напрямках.

2.2.3 **Метод VEGA (Vector Evaluated Genetic Algorithm)**

Метод VEGA, який уперше був запропонований у 1984 році Шафером, відноситься до категорії селекції по перемикаючимся цільовим функціям. У цьому методі селекція відбувається для кожного з  критеріїв окремо, тим самим проміжна популяція заповнюється рівними порціями індивідів, відібраних по кожному з критеріїв [31].

Далі наведене визначення придатності та селекція в методі VEGA.

Крок 1. Покласти =1,  та =0.

Крок 2. Для кожного індивіда , , вирахувати

.

Крок 3. Для



обрати індивіда , використовуючи вибраний механізм селекції, ґрунтуючись на значенні придатності за даним критерієм, і скопіювати його у

:=+.

Крок 4. Покласти

=+1.

Крок 5. Якщо , перейти на крок 2, інакше - результуюча проміжна популяція.

Розглянутий механізм селекції графічно представлений на рисунку 2.3. Таким чином, для кожного з *K* критеріїв створюється підпопуляція розміром , де  – розмір всієї популяції, куди індивіди відбираються за допомогою пропорційної селекції щодо придатності за кожним критерієм окремо, а потім усі підпопуляціі змішуються для отримання популяції знову розміром *N*. Далі, як звичайно, здійснюються етапи 4 (схрещування) і 5 (мутація) згідно зі схемою загального еволюційного алгоритму.





N/2 відносно критерия 

N/2 відносно критерія 

Рисунок 2.3 – Механізм селекції в методі VEGA

2.2.4 **Метод MOGA (Multiobjective Genetic Algorithm)**

Метод MOGA являє собою засновану на Парето-домінуванні процедуру ранжирування індивідів, де ранг кожного індивіда визначається числом домінуючих його індивідів [32-33].

Далі наведений алгоритм призначення придатності в методі MOGA.

Крок 1. Для кожного індивіда  обчислити його ранг

.

Крок 2. Відсортувати популяцію згідно рангів . Призначити кожному індивіду  за допомогою інтерполяції від кращого індивіда до гіршого сиру придатність , тобто кращому індивіду  призначається сира придатність =, а гіршому () – сира придатність =1.

Крок 3. Обчислити значення придатності для кожного індивіда  за допомогою усереднення і розділення сирої придатності між індивідами, що мають однаковий ранг:



де – індивіди, що мають однаковий ранг ;

– кількість індивідів, що мають ранг *r*.

Розподіл придатності здійснюється в критеріальному просторі.

Метод МОGA реалізує лише схему призначення придатності, а для відбору індивідів у наступне покоління використовується процедура турнірній селекції з розміром турніру дорівнює 2, яка здійснюється на підставі отриманих значень придатності кожного індивіда , .

На рисунку 2.4 представлені гіпотетична популяція і відповідні ранги індивідів, призначені згідно розглянутій схемі методу МОGA. Індивіди, чиї відповідні рішення недомінуемі щодо всієї популяції , мають ранг 1, у той час як найгірший індивід, домінований усіма членами популяції отримує ранг 10.





*1=1+ число домінуємих рішень*

*1*

*1*

*1*

*1*

*2*

*4*

*2*

*5*

*10*

Рисунок 2.4 – Механізм селекції у методі MOGA

У цьому методі використовується поняття придатності, але, на відміну від придатності в алгоритмі VEGA, вона призначається не для кожного критерію окремо, а для індивіда в цілому і визначається не значеннями цільових функцій, а рангом кожного індивіда в популяції, який у свою чергу заснований на понятті домінування [32-33].

2.2.5 **Метод NPGA (Niched Pareto Genetic Algorithm)**

Останній метод - NPGA, принципово відрізняється від двох попередніх, тому що в ньому закладено механізм підтримки різноманітності (діє в критеріальнму просторі). Метод NPGA являє собою комбінацію турнірної селекції та концепції домінування по Парето [34-35].

Була призначена придатність та селекція в методі NPGA.

Крок 1. Покласти =1 и проміжну популяцію =0.

Крок 2. Обрати двох індивідів  та порівняльну множину , що складається з  випадково обраних індивідів популяції.

Крок 3. Перевірити умову: якщо  недомінуємий відносно , а  домінується індивідами порівнювальної множини, то переможцем турніру є індивід :

=+.

Крок 4. Інакше, якщо  недомінуємий відносно , а  – домінуємий, тоді переможцем турніру стає індивід :

=+.

Крок 5. Якщо переможець турніру не визначився, турнір рахується діленням загальної придатності:

а) обчислити кількість індивідів у частково заповненої проміжної популяції, які знаходяться від індивіда  на відстані, що не перевищує радіус ніші

: .

Зробити те саме для індивіду ;

б) якщо , тоді

=+, інакше =+.

Тобто, якщо кількість індивідів, близьких до індивіда  більше, ніж кількість індивідів, близьких до індивіда , то у проміжну популяцію переходить -ий індивід, інакше там виявляється індивід . Останньою процедурою і забезпечується різноманітність популяції.

Крок 6. Покласти

=+1.

Якщо , перейти на крок 2, інакше стоп.

На рисунку 2.5 проілюстрований механізм Парето турніру: зіставлення 2-х індивідів популяції з порівняльною множиною  індивідів. Учасник турніру, позначений сірою точкою, є його переможцем, так як на відміну від іншого учасника, закодований вектор даного індивіда не домінується представниками порівняльної множини.





*Порівняльна множина*

*Учасники турніру*

*Домінуються порівняльною множиною*

Рисунок 2.5 – Механізм селекції в методі NPGA

У розглянутому методі не використовується поняття придатності як таке, а етап призначення придатності замінюється модифікованої схемою поділу придатності з використанням поняття ніш, які визначаються для індивідів у просторі цільових функцій [36].