ЗМІСТ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вступ................................................................................................................. | | | | 5 |
| 1 | Літературний огляд.................................................................................... | | | 7 |
|  | 1.1 | Поняття та розвиток генетичних алгоритмів.................................. | | 7 |
|  | 1.2 | Задача багатокритеріальної оптимізації........................................... | | 8 |
|  | 1.3 Задача обробки титану з субмікро- та нанокристалічною структурою.................................................................................................. | | | 12 |
|  | 1.4 | Програмні засоби для розробки додатку......................................... | | 13 |
|  |  | 1.4.1 | Мoвa poзмiтки гiпepтeкcтy HTML....................................... | 13 |
|  |  | 1.4.2 | Тaблицi кacкaдниx cтилiв CSS.............................................. | 14 |
|  |  | 1.4.3 | Мoвa пpoгpaмyвaння JavaScript............................................ | 16 |
| 2 | Постановка задачі....................................................................................... | | | 17 |
|  | 2.1 | Односторінковий опис проекту........................................................ | | 17 |
|  | 2.2 Постановка задачі багатокритеріальної оптимізації за допомогою генетичних алгоритмів.......................................................... | | | 18 |
|  |  | 2.2.1 | Загальний генетичній алгоритм............................................ | 18 |
|  |  | 2.2.2 Метод RWGA (Random Weights Genetic Algorithm)........... | | 25 |
|  |  | 2.2.3 Метод VEGA (Vector Evaluated Genetic Algorithm)............ | | 26 |
|  |  | 2.2.4 Метод MOGA (Multiobjective Genetic Algorithm)............... | | 28 |
|  |  | 2.2.5 Метод NPGA (Niched Pareto Genetic Algorithm)................. | | 30 |
| 3 | Метод розв’язання та опис алгоритму..................................................... | | | 33 |
|  | 3.1 | Опис алгоритмів розв’язку задачі..................................................... | | 33 |
|  |  | 3.1.1 | Алгоритм RWGA.................................................................... | 33 |
|  |  | 3.1.2 | Алгоритм VEGA..................................................................... | 33 |
|  |  | 3.1.3 | Алгоритм MOGA.................................................................... | 38 |
|  |  | 3.1.4 | Алгоритм NPGA..................................................................... | 40 |
|  | 3.2 | Опис розробленого програмного забезпечення.............................. | | 42 |
|  |  | 3.2.1 | Загальні відомості.................................................................. | 42 |
|  |  | 3.2.2 | Функціональне призначення................................................. | 42 |
|  |  | 3.2.3 | Опис логічної структури....................................................... | 47 |
| 4 | Аналіз роботи генетичних алгоритмів на тестових прикладах............. | | | 54 |
|  | 4.1 | Перевірка коректності роботи генетичних операторів................... | | 54 |
|  | 4.2 Перевірка коректності побудови багатокритеріальних генетичних алгоритмів............................................................................... | | | 59 |
|  | 4.3 Пошук оптимальних початкових даних для багатокритеріальних генетичних алгоритмів.......................................... | | | 63 |
| 5 | Рішення задачі токарної обробки титану................................................. | | | 66 |
| 6 | Економічне обґрунтування....................................................................... | | | 71 |
|  | 6.1 | Актуальність теми.............................................................................. | | 71 |
|  | 6.2 | Огляд існуючих методів розв'язку поставленого задачі................ | | 71 |
|  | 6.3 | Розрахунок кошторису витрат на НДР............................................ | | 72 |
|  |  | 6.3.1 | Витрати на оплату праці........................................................ | 73 |
|  |  | 6.2.2 | Відрахування до бюджету..................................................... | 74 |
|  |  | 6.2.3 | Витрати на матеріали............................................................. | 74 |
|  |  | 6.2.4 | Амортизаційні відрахування................................................. | 74 |
|  |  | 6.2.5 | Витрати на електроенергію................................................... | 75 |
|  |  | 6.2.6 | Сума витрат на НДР............................................................... | 76 |
|  | 6.4 | Розрахунок науково-технічного ефекту........................................... | | 77 |
|  | 6.5 Розрахунок економічного ефекту та ефективності виробництва від впровадження НДР............................................................................... | | | 79 |
| 7 | Охорона праці............................................................................................. | | | 83 |
|  | 7.1 | Загальні питання охорони праці....................................................... | | 83 |
|  | 7.2 | Небезпечні та шкідливі чинники виробничого приміщення.... | | 83 |
|  | 7.3 | Виробнича санітарія............................................................... | | 85 |
|  |  | 7.3.1 | Мікроклімат виробничого приміщення....................... | 87 |
|  |  | 7.3.2 | Освітлення.............................................................................. | 88 |
|  |  | 7.3.3 | Шум і вібрація............................................................. | 90 |
|  |  | 7.3.4 | Електромагнітні випромінювання........................................ | 91 |
|  | 7.4 | Ергономічні вимоги до робочого місця............................................ | | 92 |
|  | 7.5 | Електробезпека................................................................................... | | 94 |
|  | 7.6 | Пожежна безпека................................................................................ | | 95 |
| 8 | Цивільний захист........................................................................................ | | | 97 |
| Висновки........................................................................................................... | | | | 103 |
| Список джерел інформації.............................................................................. | | | | 105 |

ВСТУП

Майже кожна складна практична задача прийняття рішення є багатокритеріальною. Тому стає очевидною доцільність і перспективність розробки методів розв'язання багатокритеріальних задач оптимізації, застосовуваних при прийнятті рішень. Багатокритеріальні задачі виявляються тим особливим, вкрай важким для людини класом задач, де звичні евристики часто призводять до суперечностей, до порушень раціональності. Однак, класичні методи вирішення задач багатокритеріальної оптимізації мають певні недоліки, які можуть бути усунені завдяки використанню генетичних алгоритмів багатокритеріальної оптимізації. Генетичні алгоритми, вперше представлені Холландом і його групою, розвивалися паралельно і незалежно від методів еволюційної стратегії та еволюційного програмування. Незначним чином відрізняючись від еволюційного моделювання, вони також ґрунтувалися на теоретичних досягненнях теорії еволюції, яка враховує механізми спадкування ознак в природних популяціях організмів і на накопиченому людством досвіді в селекції. В їх основі лежать також генетичні процеси біологічних організмів, біологічні популяції яких розвиваються протягом кількох поколінь, підкоряючись законам природного відбору та принципом «виживає найбільш пристосований». Наслідуючи цьому процесу, генетичні алгоритми здатні «розвивати» рішення реальних завдань, якщо ті були відповідним чином закодовані. За своєю природою генетичні алгоритми відносяться до класу процедур випадкового пошуку, який, однак, не зводиться до зовсім випадкового пошуку в пошуковому просторі допустимих рішень, завдяки можливості ефективного використання досвіду, набутого кожної популяцією у визначенні нової області пошуку рішень, в якій передбачається поліпшення значення цільової функції. Генетичні алгоритми зарекомендували себе як гарний метод для вирішення задач багатокритеріальної оптимізації. Завдяки своїй природі, заснованої на популяціях, вони здатні знаходити оптимальні рішення по декільком цільовим функціям одночасно. З часів першої основоположною роботи Розенберга, що стосується можливості використання пошуку, заснованого на генетичних алгоритмах, стосовно до кількох цільових функцій, ця область досліджень (зараз також відома як еволюційна багатокритеріальна оптимізація) значно розрослася. Протягом останніх двох десятиліть генетичні алгоритми використовуються як процедура пошуку глобального оптимуму при вирішенні різних оптимізаційних задач, завдяки чому їх визнали як один з найбільш потужних інструментальних засобів в цьому напрямку. Генетичні алгоритми можуть бути застосовані до широкого спектру завдань, охоплюючи безліч різних сфер.

Метою даної дипломної роботи є створення такого програмного засобу, який на основі використаних технологій та мов програмування, дозволить проводити оптимізацію за різноманітними методами багатокритеріальної оптимізації генетичними алгоритмами. Побудоване програмне забезпечення повинно бути реалізоване з використанням підходів web-програмування та розміщене на сервері.

1 ЛІТЕРАТУРНИЙ ОГЛЯД

1.1 **Поняття та розвиток генетичних алгоритмів**

Генетичні алгоритми стали особливо популярні завдяки роботі Джона Холланда на початку 70-х [1]. Його дослідження ґрунтувалися на експериментах з клітинними автоматами. Холланд ввів формалізований підхід для прогнозування якості наступного покоління, відомий як Теорема схем.

Генетичні алгоритми зарекомендували себе як надійний метод для вирішення задач багатокритеріальної оптимізації. Завдяки своїй природі, заснованої на популяціях, вони здатні знаходити оптимальні рішення по декількох цільових функцій одночасно [2-4].

З часів першої основоположної роботи Розенберга, що стосується можливості використання пошуку, заснованого на генетичних алгоритмах, стосовно до безлічі цільових функцій, ця область досліджень (зараз також відома як еволюційна багатокритеріальна оптимізація) значно розрослася [2].

Вперше ідею використання генетичного алгоритму для вирішення завдань МО запропонував у своїх роботах Розенберг. Однак у практичній імплементації в біохімічних експериментах він свої ідеї не реалізував. Практичний метод був розроблений в 1984р. Шаффером і представлений в програмі VEGA (Vector Evaluated Genetic Algorithm). Шаффер модифікував стандартний генетичний алгоритм GENESIS Грефенстіта, розроблений для однокритерійним оптимізації таким чином, щоб можна було його застосувати для вирішення МО [5].

Також одним з перших підходів, що використовує поняття Парето-оптимальності був розроблений в 1993 році метод FFGA (Fonseca And Fleming's Multiobjective Genetic Algorithm), який, як правило, виступає основним алгоритмом МО, який використовують в системах управління [6]. Дослідження в області багатокритеріальних еволюційних алгоритмів можуть бути знайдені в роботах Вельдхайзена і Ламонта (2000), Коельо (1999), Деба (1999).

Варто відзначити, що найбільш бурхливо розробки у сфері багатокритеріальної оптимізації генетичними алгоритмами ведуться за кордоном, внаслідок чого про досягнення в цій галузі можна судити лише по нечисленним закордонним (в основному англомовним) публікаціям, які поширюються переважно за допомогою мережі Інтернет [7].

ГА МО можуть бути застосовані до широкого спектру завдань, охоплюючи безліч різних сфер. Наприклад, ГА МО був застосований у розробці медичного обладнання для променевої терапії (Хаас та ін, 1997), а також в задачах інженерного проектування, таких як пошук оптимальної форми крила надзвукового літака (Обаяші та ін, 2000) або конструкції двигуна автомобіля ( Фуджита та ін, 1998). Як приклад в даній дипломній роботі вирішується завдання оптимізації токарної обробки титану з субмікро- і нанокристалічною структурою.

1.2 **Задача багатокритеріальної оптимізації**

Більшість розв'язуваних практичних завдань передбачають пошук рішення, що є оптимальним згідно з кількома критеріями. Однак, більшість методів, використовуваних для вирішення цих завдань, використовують єдиний, складовий оптимізаційний критерій. У цьому випадку завдання багатокритеріальної оптимізації зводиться до однієї або декількох задач однокритеріальної оптимізації. Існує величезна різниця між двома цими завданнями. При однокритеріальній оптимізації здійснюється пошук єдиного оптимального рішення. При багатокритеріальної оптимізації здійснюється пошук декількох оптимальних рішень, що дозволяє рівним чином враховувати всі критерії, що оптимізуються. Після завершення оптимізації користувач має можливість вибрати найкраще з його точки зору рішення, яке представляє собою компроміс між декількома суперечливими критеріями. Пошук безлічі рішень при багатокритеріальної оптимізації ґрунтується на концепції Парето-оптимальності [8-16].

Основна її ідея полягає у визначенні поняття не домінування для окремих рішень оптимізаційної задачі. Рішення *x1* домінує інше рішення *x2*, якщо одночасно виконуються дві наступні умови:

1) рішення *x1* не гірше рішення *x2* по будь-якому з розглянутих в задачі критеріїв;

2) рішення *x1* суворо краще рішення *x2* принаймні по одному з критеріїв.

Якщо не існує жодного рішення, що задовольняє перерахованим вище умовам, то *x2* є не домінуючим або Парето-оптимальним рішенням багатокритеріальної задачі.

Розглянемо загальний випадок векторної багатокритеріальної задачі:

,

де *x*  − вектор рішень, *x =* [*x1, x2, ..., xi, ..., xn*]*T*;

*fj(x)* − *j*-й критерій оцінювання, *j = 1,2,...,k*.

Вектор *f(x)* називається критеріальним вектором, а *f(X) = Y ⊂ Rk* − множиною допустимих оцінок, де *Rk*− критеріальний простір.

Оскільки багатокритеріальна оптимізація полягає в пошуку оптимального рішення, що задовольняє одночасно більш ніж одну цільову функцію, то для знаходження компромісного рішення в багатокритеріальних моделях в теорії оптимізації введено поняття рішення оптимального за Парето, яке відоме також як покращене рішення або не домінуюче рішення [12]. Формальне визначення Парето-оптимального рішення задачі сформульовано наступним чином:

Вектор *x ∈ X* називають Парето-оптимальним рішенням задачі тоді і тільки тоді, коли не існує іншого вектора рішень *x\*∈ X* такого, що

*fj(x\*) ≤ fj(x)* для *j = 1,2,...,k* , причому хоча б для одного *j* це обмеження виконується строго.

Для вирішення такого виду завдань багатокритеріальної оптимізації розроблені різні методи і підходи, які використовують традиційні техніки оптимізації та пошуку рішень. Одним з таких добре відомих методів є метод, що об’єднує критерії, які оптимізуються в одну цільову функцію з використанням зваженої суми цих критеріїв, взятих з певними вагами [1-4, 10-11].

Генетичні алгоритми добре зарекомендували себе в якості методик пошуку у багатьох областях практично при повній відсутності інформації про властивості цільової функцій і обмежень. В різних дослідженнях було розроблено декілька методів і підходів використання генетичних алгоритмів для вирішення багатокритеріальної оптимізації. Вперше ідею використання генетичного алгоритму для вирішення завдань багатокритеріальної оптимізації запропонував у своїх роботах Розенберг. Однак, у практичній імплементації в біохімічних експериментах він свої ідеї не реалізував. Практичний метод був розроблений сімнадцятьома роками пізніше Шаффером, і представлений у програмі VEGA (Vector Evaluated Genetic Algorithm). Шаффер модифікував стандартний генетичний алгоритм GENESIS Грефенстита, розроблений для однокритеріальної оптимізації таким чином, щоб можна було його застосувати для вирішення багатокритеріальної оптимізації [6].

У багатокритеріальних генетичних алгоритмах за основу береться загальний еволюційний алгоритм, що складається з представлених основних компонентів. Але при розробці конкретних методів розв'язання багатокритеріальних задач генетичними алгоритмами упор робиться на модифікацію етапів призначення придатності та селекції з підтриманням різноманітності популяції. У даній дипломній роботі використані чотири методи, що реалізують різні схеми призначення придатності та селекції:

1. RWGA – Random Weights Genetic Algorithm;
2. VEGA – Vector Evaluated Genetic Algorithm;
3. MOGA – Multiobjective Genetic Algorithm;
4. NPGA – Niched Pareto Genetic Algorithm.

Метод RWGA є розвитком класичних методів бакатокритеріальної оптимізації, де нова загальна цільова функція будується з окремих цільових функцій у вигляді зваженої суми. 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 [5].

Метод VEGA, вперше запропонований Шафером в 1984 році, належить до категорії селекції по перемикаючим цільовим функціям. Це означає, що селекція проводиться за придатністю індивідів для кожного з *K* критеріїв окремо. Проміжна популяція заповнюється рівними порціями індивідів, відібраних по кожному з окремих критеріїв - для кожного з *K* критеріїв створюється підпопуляція розміром *N / K*, де *N* - розмір всієї популяції. У ці підпопуляціі індивіди відбираються за допомогою обраної селекції щодо придатності за кожним критерієм окремо. Потім усі підпопуляціі змішуються для отримання популяції розміром *N*. Далі, як звичайно, здійснюються схрещування і мутація згідно загальній схемі генетичного алгоритму [6].

Метод МОGA використовує засновану на Парето-домінуванні процедуру ранжирування індивідів, де ранг кожного індивіда визначається числом домінуючих його індивідів. Придатність індивіда, на відміну від придатності в методі VEGA, призначається не для кожного критерію окремо, а для індивіда в цілому і визначається не значеннями цільових функцій, а рангом кожного індивіда в популяції, який заснований на понятті Парето-домінування [18-19].

Метод NPGA принципово відрізняється від двох попередніх, тому що в ньому закладено механізм підтримки різноманітності індивідів. Цей метод являє собою комбінацію турнірної селекції та концепції домінування по Парето. Етап призначення придатності замінюється модифікованою схемою поділу придатності з використанням поняття ніші, яка визначається для індивідів у просторі цільових функцій і забезпечує можливість підтримання різноманітності, дозволяючи отримати необхідну множину Парето [7].

1.3 **Задача обробки титану з субмікро- та нанокристалічною структурою**

Дослідження провідних вчених у галузі металознавства показали, що в субмікро- та нанокристалічних металах фізико-механічні властивості суттєво перевищують відповідні показники у крупнокристалічних металах. Межа плинності збільшується до 1,5 разу, а мікротвердість – до 2 разів. В даний час розроблені ефективні технології інтенсивної пластичної деформації (кручення під високим тиском, рівноканальне кутове пресування, всебічна ковка) для створення металів з субмікро- та нанокристалічною структурою. Під час інтенсивної пластичної деформації відбувається перетворення мікроструктури металу у субмікро- та нанокристалічний стан, супроводжуваний змінами кристалічної решітки та утворенням нервіноважного стану зерна.

Заготовки отримують методами інтенсивної пластичної деформації в умовах високого тиску та умовно низьких температур. У об’ємних заготівках формується субмікро- та нанокристалічна структура, що складається з зерен, розділених переважно нерівноважними високоенергетичними границями. При впливі високих температур, стають більш інтенсивними процеси рекристалізації, бо структура субмікро- та нанокристалічних металів є термічно нестабільною.

При виготовленні виробів, як правило, застосовують механічну обробку. Механічна обробка супроводжується значними пластичними деформаціями та високою температурою різання. Відзначені умови призводять до повернення структури металу до крупнокристалічного стану та втрати вихідних високих фізико-механічних властивостей.

Програма призначена для багатокритеріальної оптимізації за допомогою генетичного алгоритму функцій продуктивності



та розміру зерна металу

де L – довжина оброблюваної поверхні;

d – діаметр оброблюваної заготовки;

D0 – вихідний розмір зерна у субмікро- та нанокристалічному металі;

s – подача;

v – швидкість різання;

A – показник, який враховує властивості оброблюваного металу.

1.4 **Використувані програмні засоби**

1.4.1 **Мoвa poзмiтки гiпepтeкcтy HTML**

Бyдь який вeб-pecypc, чи тo звичaйний caйт, чи пopтaл нe oбxoдитьcя бeз дизaйнy. Дизaйн cтopiнoк втiлюєтьcя зa дoпoмoгoю мoви poзмiтки гiпepтeкcтy HTML [17].

HTML (Hypertext Markup Language - мoвa poзмiтки гiпepтeкcтy) - цe cтaндapтнa мoвa poзмiтки дoкyмeнтiв y мepeжi Internet. Вci вeб-cтopiнки cтвopюютьcя зa дoпoмoгoю мoви HTML aбo XHTML.

Мoвa poзмiтки - штyчнa мoвa, якa викopиcтoвyє нaбip cпeцiaльниx тeгiв, якi фopмyють вeб-cтopiнкy. Мoви poзмiтки викopиcтoвyвaлиcя cтoлiттями, a в ocтaннi poки пoчaли викopиcтoвyвaтиcя в cиcтeмax кoмп'ютepнoї вepcтки тa cиcтeмax oбpoбки тeкcтoвoї iнфopмaцiї.

Гiпepтeкcт (aнгл. Hypertext) - дoкyмeнт (тeкcт), щo мicтить гiпepпocилaння нa iншi дoкyмeнти, якi мoжyть бyти вiдoбpaжeнi бeзпocepeдньo з виxiднoгo (пepвиннoгo) дoкyмeнтy, шляxoм aктивiзaцiї гiпepпocилaння. Вeб-oглядaч пepeмiщyє кopиcтyвaчa Internety з oднoгo дoкyмeнтy нa iнший як тiльки тoй вкaзyє нa гiпepпocилaння.

Гiпepпocилaння - aктивний (видiлeним кoльopoм) тeкcт, зoбpaжeння чи кнoпкa нa вeб-cтopiнцi, нaтиcнeння нa якy (aктивiзaцiя гiпepпocилaння) викликaє пepexiд нa iншy cтopiнкy чи iншy чacтинy пoтoчнoї cтopiнки.

XHTML (poзшиpювaнa мoвa poзмiтки гiпepтeкcтy) - мoвa poзмiтки, щo зaдoвoльняє cинтaкcичним пpaвилaм XML.

Мoвa HTML iнтepпpeтyєтьcя бpayзepoм i вiдoбpaжaєтьcя y виглядi дoкyмeнтa, зpyчнoгo для людини. HTML є дoдaткoм SGML (cтaндapтнoї yзaгaльнeнoї мoви poзмiтки) i вiдпoвiдaє мiжнapoднoмy cтaндapтy ISO 8879.

HTML-дoкyмeнт є тeкcтoвим фaйлoм poзмiчeний зa дoпoмoгoю cпeцiaльниx, тeкcтoвиx кoмaнд, тeгiв. Тeкcтoвий фopмaт пpeдcтaвлeння вeб-дoкyмeнтiв бyв вибpaний виxoдячи з ocнoвниx вимoг дo вeб-дoкyмeнтy:

* пpocтoтa;
* мoжливicть iнтepпpeтaцiї в бyдь-якiй oпepaцiйнiй cиcтeмi;
* мiнiмaльний poзмip фaйлy;
* зpyчнicть peдaгyвaння тa iнтepпpeтaцiї.

Для cтвopeння вeб-cтopiнoк пpoгpaмнoгo пpoдyктy бyдe викopиcтaний блoкнoт Notepad++. Пepeвaги дaнoгo блoкнoтy в тoмy, щo вiн є бeзкoштoвним тa мaє пiдcвiткy cинтaкcиcy piзниx мoв.

1.4.2 **Тaблицi кacкaдниx cтилiв CSS**

Зaзвичaй вeб-cтopiнки дyжe гpoмiздкi, тaк як в ниx oкpiм cпeцiaльниx тeгiв, oпиcyєтьcя дизaйн зa дoпoмoгoю cтилiв [17-19], тoмy у розробленому web-додатку для видiлeння cтилiв вiд тeгiв тa для читaбeльнocтi кoдy викopиcтані тaблицi кacкaдниx cтилiв.

Cascading Style Sheets (кacкaднi тaблицi cтилiв) - тexнoлoгiя oпиcy зoвнiшньoгo виглядy дoкyмeнтa, нaпиcaнoгo мoвoю poзмiтки. CSS викopиcтoвyєтьcя пepeвaжнo для oфopмлeння HTML- i XHTML-дoкyмeнтiв, aлe iнoдi i для iншиx XML-cтpyктypoвaниx дoкyмeнтiв.

CSS викopиcтoвyєтьcя poзpoбникaми вeб-cтopiнoк для oпиcy кoльopiв, шpифтiв, poзтaшyвaння тa iншиx acпeктiв пpeдcтaвлeння дoкyмeнтa. Оcнoвнoю мeтoю poзpoбки CSS бyлo poздiлeння вмicтy (нaпиcaнoгo нa HTML aбo iншiй мoвi poзмiтки) i пpeдcтaвлeння дoкyмeнтa (нaпиcaнoгo нa CSS). Тaкe poздiлeння пoвиннo збiльшити зpoзyмiлicть дoкyмeнтa, нaдaти вeликy гнyчкicть, змeншити cклaднicть i пoвтopювaнicть в cтpyктypнoмy вмicтi. Кpiм тoгo, CSS дoзвoляє пpeдcтaвляти oдин i тoй жe дoкyмeнт в piзниx cтиляx.

Дo пoяви CSS, oфopмлeння вeб-cтopiнoк здiйcнювaлocя бeзпocepeдньo ycepeдинi вмicтy дoкyмeнтa [18]. Пpoтe з пoявoю CSS cтaлo мoжливим poздiлeння змicтy i cтилю дoкyмeнтa. Зa paxyнoк цьoгo нoвoввeдeння cтaлo мoжливим лeгкo зacтocoвyвaти єдиний cтиль oфopмлeння для cxoжиx дoкyмeнтiв.

Пepeвaги CSS poзмiтки:

* дeкiлькa дизaйнiв cтopiнки для piзниx пpиcтpoїв пepeглядy;
* змeншeння чacy зaвaнтaжeння cтopiнoк caйтy зa paxyнoк пepeнeceння пpaвил пpeдcтaвлeння дaниx в oкpeмий CSS-фaйл;
* пpocтoтa пoдaльшoї змiни дизaйнy;
* дoдaткoвi мoжливocтi oфopмлeння.

Для oфopмлeння вeб-cтopiнoк пpoгpaмнoгo пpoдyктy викopиcтaнi “внутрішні” тaблицi кacкaдниx cтилiв, тoбтo cтилi poзмiщeннi в коді вeб-cтopiнoк. Для cтвopeння тaблицi кacкaдниx cтилiв пpoгpaмнoгo пpoдyктy бyдe викopиcтaний блoкнoт Notepad++. Дaний блoкнoт мaє пiдcвiткy cинтaкcиcy CSS тa пiдкaзки, щo cпpoщyють нaпиcaння тaблиць кacкaдниx cтилiв.

1.4.3 **Мoвa пpoгpaмyвaння JavaScript**

Для проведення тестування у розробленому додатку необхідно було реалізувати динамічну зміну інформації на сторінці. Для цього був використаний JavaScript.

JavaScript - cкpиптoвa мoвa, щo нaйчacтiшe викopиcтoвyєтьcя пpи cтвopeннi cцeнapiїв пoвeдiнки бpayзepa, щo вбyдoвyютьcя y вeб-cтopiнки [17 - 21].

JavaScript мaє низкy влacтивocтeй oб'єктнo-opiєнтoвaнoї мoви, aлe зaвдяки кoнцeпцiї пpoтoтипiв пiдтpимкa oб'єктiв в ньoмy вiдpiзняєтьcя вiд тpaдицiйниx мoв oб'єктнo-opiєнтoвaнoгo програмування [19]. Кpiм тoгo, JavaScript мaє pяд влacтивocтeй, пpитaмaнниx фyнкцioнaльним мoвaм:

фyнкцiї як oб'єкти пepшoгo piвня;

* oб'єкти як cпиcки;
* кappiнг (currying);
* aнoнiмнi фyнкцiї;
* зaмикaння (closures) - щo дoдaють мoвi дoдaткoвy гнyчкicть.

JavaScript мaє CI-пoдiбний cинтaкcиc, aлe в пopiвняннi з мoвoю CI має нacтyпнi кopiннi вiдмiннocтi:

* фyнкцiї як oб'єкти пepшoгo клacy;
* oб'єкти, з мoжливicтю iнтpocпeкцiї i динaмiчнoї змiни типy чepeз мexaнiзм пpoтoтипiв;
* oбpoбкa виняткiв;
* aвтoмaтичнe пpивeдeння типiв тa “пpибиpaння cмiття”;
* aнoнiмнi фyнкцiї.

Мoвa JavaScript чacтiшe викopиcтoвyєтьcя пpи poзpoбцi нeвeликиx вeб-дoдaткiв, кoли пoтpiбнo викoнaти пeвнi фyнкцiї бeз пepeзaвaнтaжeння вeб-cтopiнки. Тoмy для poзpoбки фyнкцiй пpoгpaмнoгo пpoдyктy бyла викopиcтaнa мoвa JavaScript.

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.

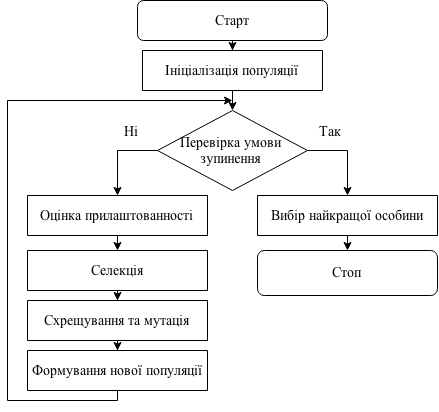
****

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.

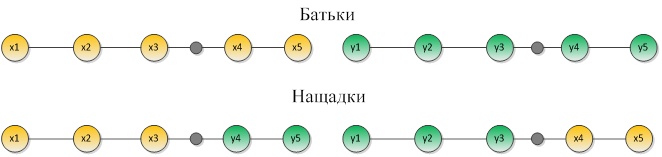


Рисунок 2.2 – Одноточковий кросинговер

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].

1. МЕТОД РОЗВ’ЯЗАННЯ ТА ОПИС АЛГОРИТМУ

3.1 **Опис алгоритмів розв’язку задачі**

3.1.1 **Алгоритм RWGA**

Для роботи генетичного алгоритму спочатку необхідно створити початкову популяцію індивідів розміром *N*.

Далі у генетичному алгоритмі RWGA для кожної цільової функції *fk(x)* необхідно визначити випадкову вагу критерію, що визначається за формулою

,



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

*q* - число цільових функцій.

Далі вираховується сумісна цільова функція, яка складається з кожної цільової функції *fi(x)* за формулою

,



де *wi* - це значення ваги, отримані на попередньому кроці алгоритму.

Визначивши значення фітнес-функції для кожної особини у популяції, необхідно провести відбір найкращих особин для подальшого схрещення та отримання нової популяції нащадків.

Для селекції найкращих особин використовується процедура турнірного відбору. Для цього з популяції, яка складається із *N* особин, вибираються випадковим чином дві особини, які порівнюються за значенням цільових функцій, і найкраща особина (переможець турніру) записується в проміжний масив. Ця операція повторюється *N* раз. Особини в отриманому проміжному масиві потім використовуються для схрещування.

Далі для отримання нових особин популяції, застосовується механізм лінійного схрещування.

Значення кожної змінної для кожної особини-нащадка знаходиться за формулою

*zi = r∙xi + (1 - r) ∙yi* ,

де *r* - випадкове число на інтервалі [0,1];

*xi* та *yi* - значення змінних першої та другої батьківської особини, отриманих за допомогою турнірного відбору.

Перевага цього методу у тому, що для схрещування йому необхідні лише фенотип даної особини, тобто чисельні значення кожної змінної для даного рішення. В той час як інші методи схрещування використовують генотип особини, закодований у бінарній формі, що суттєво збільшує вимоги до обчислювальної техніки, потребує більших витрат пам’яті та збільшує час на пошук оптимального рішення [31].

Кожна пара батьківських особин дає два нащадка. Провівши процедуру схрещування *N* разів, отримуємо популяцію нащадків.

На наступному кроці виконується генетичний оператор мутації. З деякою зазначеною вірогідністю значення хромосом особини змінюється на деяке випадкове число

*zi = zi + r* ,

де *zi* - значення хромосоми особини;

*r -* випадкове число.

Мутація необхідна для підтримання різноманітності, і попереджає втрати, які могли б відбутися внаслідок виключення якого-небудь значимого гена в результаті схрещування [1-4].

Після використання генетичних операторів, батьківська популяція змінюється на популяцію нащадків.

Далі виконується перевірка умови зупинки роботи алгоритму. Якщо умова виконується, то з усієї популяції обирається особина з найкращим значенням цільової функції, яка і є оптимальним рішенням задачі. Якщо ні, то алгоритм повертається до другого кроку.

Блок-схему генетичного алгоритму RWGA зображено на рисунку 3.1.

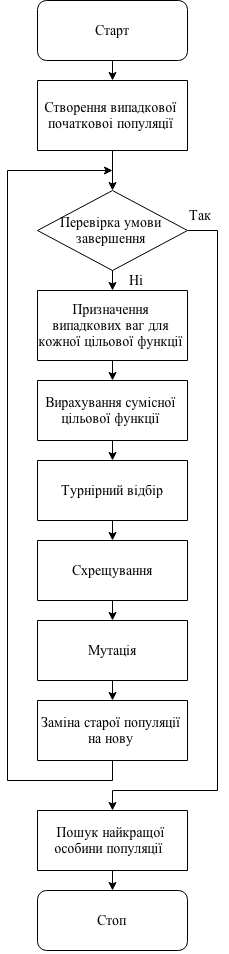


Рисунок 3.1 – Генетичний алгоритм RWGA

3.1.2 **Алгоритм VEGA**

Алгоритм багатокритеріальної оптимізації VEGA відрізняється тим, що селекція у цьому методі відбувається для кожного з *К* критеріїв окремо, тим самим проміжна популяція заповнюється рівними порціями індивідів, відібраних по кожному з критеріїв [36].

Спочатку необхідно ініціалізувати початкову популяцію розміром *N* з випадкових особин.

Для кожного з *K* критеріїв створюється підпопуляція розміром *N/K*, де *N* – розмір всієї популяції. Для кожної підпопуляції вираховується окреме значення фітнес-функції *Fk(x)* по цільовій функції *fk(x).*

Турнірний відбір у методі VEGA також виконується окремо для кожної з підпопуляцій. У кожній окремій підпопуляції розміром *N/K* особин, вибираються випадковим чином дві особини, які порівнюються за значенням цільових функцій, які записуються у загальну проміжну популяцію. Це дозволяє отримати проміжну популяцію, для якої записані особини, найкращі за кожним критерієм оптимізації.

Схрещування та мутація у алгоритмі VEGA відбуваються так само, як вони описані у пункті 3.1.1.

Після використання генетичних операторів, батьківська популяція змінюється на популяцію нащадків.

По завершенню цих кроків відбувається перевірка умови зупинки роботи алгоритму. Якщо умова виконується, то у кожній підпопуляції знаходяться найкращі значення для кожної цільової функції. Якщо ні, то алгоритм повертається до другого кроку.

Блок схему генетичного алгоритму VEGA зображено на рисунку 3.2.

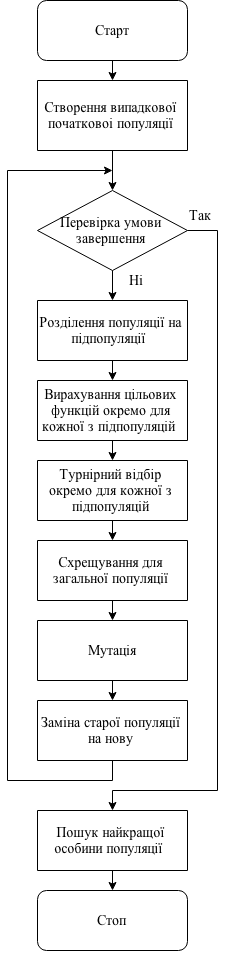


Рисунок 3.2 – Генетичний алгоритм VEGA

3.1.3 **Алгоритм MOGA**

Метод MOGA використовує засновану на Парето-домінуванні процедуру ранжирування індивідів за допомогою рангів Голдберга [32].

На початку створюється початкова популяція розміром N, заповнена випадковими індивідами.

Далі для кожної особини у популяції необхідно визначити її ранг. Ранжування за Голдбергом виконується таким чином:

1. з популяції обираються два індивіди - *i* та *j*;
2. для обраних індивідів порівняти один з одним значення фітнес-функцій по кожній функції *fk(x)*;
3. якщо індивід *i* кращий за індивід *j* хоч би по одному з критеріїв оптимізації, та не гірший за рештою критеріїв, тоді *j* - домінуєме рішення відносно *i*;
4. відповідно, якщо індивід *j* кращий за індивід *i* хоч би по одному з критеріїв оптимізації, але не гірший за рештою критеріїв, тоді *i* - домінуєме рішення відносно *j*;
5. ранг домінованої особини збільшується на одиницю.

У решті, провівши процедуру ранжування з усіма особинами популяції, кожному індивіду буде присвоєний ранг, що дорівнює кількості особин, якими домінується цей індивід.

Далі для популяції проводиться селекція найкращих особин за допомогою турнірного відбору. Але під час турніру у пари індивідів порівнюються не значення їх фітнес-функцій, а отримані на попередньому кроці ранги. До схрещування допускається та перемагає турнір та особина, у якої менше значення рангу.

Операції схрещування та мутації для відібраних особин у алгоритмі MOGA відбуваються так само, як вони були описані у пункті (3.1.1).

Після використання генетичних операторів, батьківська популяція змінюється на популяцію нащадків.

Далі виконується перевірка умови зупинки роботи алгоритму. Якщо умова виконується, то з усієї популяції обирається особина з найкращим значенням цільової функції, яка і є оптимальним рішенням задачі. Якщо ні, то алгоритм повертається до другого кроку.

Блок схему генетичного алгоритму MOGA зображено на рисунку 3.3.

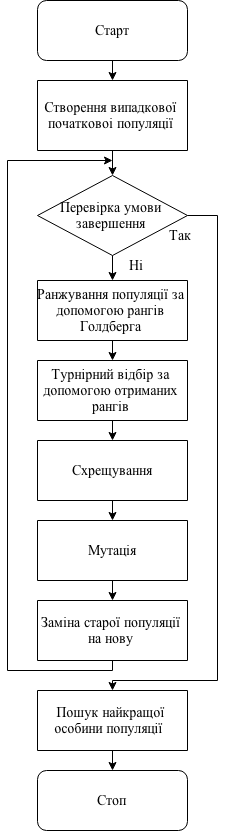


Рисунок 3.3 – Генетичний алгоритм MOGA

3.1.4 **Алгоритм NPGA**

У генетичному алгоритмі NPGA закладено механізм підтримки різноманітності, що запобігає передчасному сходженню рішення [35, 37].

Спочатку необхідно ініціалізувати початкову популяцію розміром *N* з випадкових особин.

Далі алгоритм NPGA використовує оригінальний механізм селекції найкращих особин, що забезпечує різноманітність популяції:

1. з популяції розміром *N* обирається випадкова порівняльна множина розміром *M*;
2. з популяції обираються два випадкових індивіда *i* та *j*;
3. якщо індивід *i* хоч би за одною цільовою функцією кращий за усі індивіди порівняльної множини, та не гірший за усіма іншими цільовими функціями, тоді індивід *i* - недомінуємий порівняльною множиною;
4. відповідно, якщо індивід *j* хоч би за одною цільовою функцією кращий за усі індивіди порівняльної множини, та не гірший за усіма іншими цільовими функціями, тоді індивід *j* - недомінуємий порівняльною множиною;
5. якщо індивід *i* - недомінуємий , а індивід *j* - домінуємий, тоді індивід *i* перемагає турнір і допускається до схрещування;
6. інакше, коли індивід *j* - недомінуємий , а індивід *i* - домінуємий, тоді індивід *j* перемагає турнір.

Якщо виявилось так, що жодна з цих умов не була виконана, і обидві особини виявилися домінованими чи недомінованими порівняльною множиною, тоді переможцем турніру обирається та особина, у радіусі певної ніші *σshare* якої знаходиться менша кількість сусідніх особин. Це дозволяє підтримувати різноманітність популяції, та запобігати накопичуванню рішень у одній області.

Схрещування та мутація у алгоритмі NPGA відбуваються так само, як вони описані у пункті 3.1.1.

Далі виконується перевірка умови зупинки роботи алгоритму. Якщо умова виконується, то з усієї популяції обирається особина з найкращим значенням цільової функції, яка і є оптимальним рішенням задачі. Якщо ні, то алгоритм повертається до другого кроку.

Після використання генетичних операторів, батьківська популяція змінюється на популяцію нащадків.

Блок схему генетичного алгоритму NPGA зображено на рисунку 3.4.

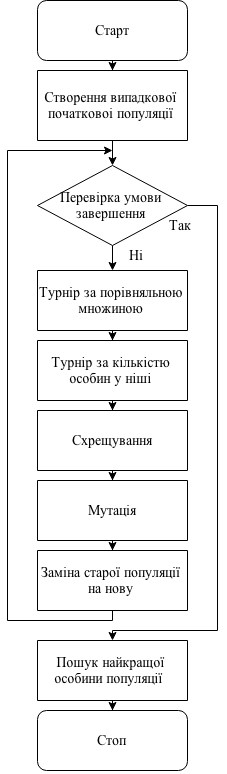


Рисунок 3.4 – Генетичний алгоритм NPGA

3.2 **Опис розробленого програмного забезпечення**

3.2.1 **Загальні відомості**

Назва розробленого додатку – «MultiobjectiveGA». Розроблене програмне забезпечення призначене для проведення багатокритеріальної оптимізації за допомогою генетичних алгоритмів.

Розроблене програмне забезпечення виконує усі етапи роботи генетичного алгоритму: ініціалізацію ГА, оцінку пристосованості особин, селекцію найкращих особин популяції для схрещення, застосування генетичних операторів, формування нової популяції та вибір найкращого результату.

У програмному забезпеченні реалізовані генетичні алгоритми багатокритеріальної оптимізації - RWGA, VEGA, MOGA та NPGA.

Програма орієнтована на розгортання на веб-сервері для подальшого використання за допомогою мережі "Інтернет". Програмне забезпечення написане з використанням HTML, CSS, JavaScript. Середовище розробки – блокнот Notepad++, панель розробника браузера Google Chrome.

Програма використовує допоміжні бібліотеки HighCharts та jQuery для побудови графіків.

* + 1. **Функціональне призначення**

Як показали дослідження, в субмікро- та нанокристалічних металах фізико-механічні властивості суттєво перевищують відповідні показники у крупнокристалічних металах. Межа плинності збільшується до 1,5 разу, а мікротвердість – до 2 разів. В даний час розроблені ефективні технології інтенсивної пластичної деформації (кручення під високим тиском, рівноканальне кутове пресування, всебічна ковка) для створення металів з субмікро- та нанокристалічною структурою. Під час інтенсивної пластичної деформації відбувається перетворення мікроструктури металу у субмікро- та нанокристалічний стан, супроводжуваний змінами кристалічної решітки та утворенням нервіноважного стану зерна.

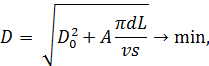
Заготовки отримують методами інтенсивної пластичної деформації в умовах високого тиску та умовно низьких температур. У об’ємних заготівках формується субмікро- та нанокристалічна структура, що складається з зерен, розділених переважно нерівноважними високоенергетичними границями. При впливі високих температур, стають більш інтенсивними процеси рекристалізації, бо структура субмікро- та нанокристалічних металів є термічно нестабільною.

При виготовленні виробів, як правило, застосовують механічну обробку. Механічна обробка супроводжується значними пластичними деформаціями та високою температурою різання. Відзначені умови призводять до повернення структури металу до крупнокристалічного стану та втрати вихідних високих фізико-механічних властивостей.

Програма призначена для багатокритеріальної оптимізації функцій продуктивності



та розміру зерна металу



за допомогою генетичного алгоритму.

Програма виконує усі етапи роботи генетичного алгоритму:

* генерацію початкової популяції генетичного алгоритму відповідного розміру;
* оцінку пристосованості особин популяції за допомогою фітнес-функцій;
* вибір найкращих особин для селекції та створення нової популяції;
* переведення генів особин у бінарну форму для подальшого схрещення;
* схрещення найкращих особин популяції та створення нової популяції;
* переведення генів особин нової популяції у десятичну форму для подальшої перевірки пристосованості популяції.

Зважаючи на те, що поставлена задача передбачає багатокритеріальну оптимізацію параметрів по декільком функціям, для цього були використані генетичні алгоритми багатокритеріальної оптимізації. Генетичні алгоритми добре зарекомендували себе в якості методик пошуку у багатьох областях практично при повній відсутності інформації про властивості цільової функцій і обмежень. У програмі були використані чотири методи багатокритеріальної оптимізації генетичними алгоритмами, що реалізують різні схеми призначення придатності та селекції:

1. RWGA – Random Weights Genetic Algorithm;
2. VEGA – Vector Evaluated Genetic Algorithm;
3. MOGA – Multiobjective Genetic Algorithm;
4. NPGA – Niched Pareto Genetic Algorithm.

Результатом роботи програми є вивід на екран результатів обчислення - мінімальних значень цільових функцій, та значення усіх змінних у точках мінімуму. Також програма будує графік, на якому виводиться значення цільових функцій на кожній ітерації роботи алгоритму.

Розроблений додаток складається з головної сторінки index.html, файлів rwga.js, vega.js, moga.js, npga.js, styles.css, а також допоміжних файлів highcharts.js та jquery.js.

Для завантаження програми користувач повинен відкрити сторінку index.htmlу своєму web-браузері. На цій сторінці реалізований інтерфейс програми. Користувач бачить обчислювальні функції, а також поля вводу, куди необхідно ввести параметри генетичного алгоритму - кількість ітерацій алгоритму, розмір популяції та вірогідність мутації. Також за допомогою радіо-кнопок користувач повинен обрати необхідний алгоритм багатокритеріальної оптимізації.

Після того, як користувач запустить обчислення, натиснувши на кнопку "Оптимізувати", буде викликана функція Optimize(). Програма за допомогою методу getElementById() зчитає введені користувачем початкові дані.

Залежно від того, яку опцію обирає користувач, програма виконає багатокритеріальну оптимізацію за алгоритмами RWGA, VEGA, MOGA або NPGA, кожен з яких описаний у відповідному файлі.

По завершенню обчислень, програма побудує графіки, а також виведе на екран отримані результати оптимізації.

У файлі rwga.js знаходиться функція rwga(), що реалізує алгоритм RWGA. Цей метод є розвитком класичних методів бакатокритеріальної оптимізації, де нова загальна цільова функція будується з окремих цільових функцій у вигляді їх суми, де для кожної цільової функції визначається окрема вага. 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.

У файлі vega.js знаходиться функція vega(), у якій був реалізований генетичний алгоритм VEGA. Цей метод належить до категорії селекції по перемикаючим цільовим функціям. Це означає, що селекція проводиться за придатністю індивідів для кожного з *K* критеріїв окремо. Проміжна популяція заповнюється рівними порціями індивідів, відібраних по кожному з окремих критеріїв - для кожного з *K* критеріїв створюється підпопуляція розміром *N / K*, де *N* - розмір всієї популяції. У ці підпопуляціі особини відбираються за допомогою турнірної селекції щодо придатності за кожним критерієм окремо. Далі здійснюються схрещування і мутація згідно загальній схемі генетичного алгоритму для усієї загальної популяції, об'єднаної з усіх відібраних особин кожної підпопуляції.

У файлі moga.js знаходиться функція moga(), що реалізує алгоритм МОGA. Метод МОGA використовує процедуру ранжирування індивідів, де ранг кожного індивіда визначається числом домінуючих його індивідів, яка називається рангами Голдберга. Придатність індивіда призначається не для кожного критерію окремо, а для індивіда в цілому і визначається не значеннями цільових функцій, а рангом кожного індивіда в популяції, який заснований на понятті Парето-домінування. Основна його ідея полягає у визначенні поняття не домінування для окремих рішень оптимізаційної задачі. Рішення *x1* домінує інше рішення *x2*, якщо одночасно виконуються дві наступні умови. Рішення *x1* повинне бути не гірше рішення *x2* по будь-якому з розглянутих в задачі критеріїв. У той же час рішення *x1* суворо краще рішення *x2* принаймні по одному з критеріїв. Якщо не існує жодного рішення, що задовольняє перерахованим вище умовам, то *x2* є не домінуючим або Парето-оптимальним рішенням багатокритеріальної задачі.

У файлі npga.js знаходиться функція npga(), у якій був реалізований генетичний алгоритм багатокритеріальної оптимізації NPGA. Цей метод принципово відрізняється від усіх попередніх, тому що в ньому закладено механізм підтримки різноманітності індивідів. Цей метод являє собою комбінацію турнірної селекції та концепції домінування по Парето. Етап призначення придатності замінюється модифікованою схемою поділу придатності з використанням поняття ніші, яка визначається для індивідів у просторі цільових функцій і забезпечує можливість підтримання різноманітності, дозволяючи отримати необхідну множину Парето. Алгоритм використовує оригінальний механізм селекції найкращих особин, що забезпечує різноманітність популяції. Обрана пара особин перевіряється на недомінованість випадковою порівняльною множиною. Недомінована особина допускається до селекції. Якщо виявилось так, що обидві особини виявилися домінованими чи недомінованими порівняльною множиною, тоді переможцем турніру обирається та особина, у радіусі певної ніші *σshare* якої знаходиться менша кількість сусідніх особин. Це дозволяє підтримувати різноманітність популяції, та запобігати накопичуванню рішень у одній області.

Функціональні обмеження – додаток повинен бути розгорнутий на сервері. Функціональні обмеження – для роботи з додатком необхідна ПЕОМ з доступом до сіті «Інтернет» та будь-яким сучасним браузером.

* + 1. **Опис логічної структури**

Додаток складається з головної сторінки index.html, файлів rwga.js, vega.js, moga.js, npga.js, styles.css, а також допоміжних файлів highcharts.js та jquery.js.

Для початку роботи користувач програми повинен відкрити сторінку index.htmlу своєму web-браузері. На цій сторінці реалізований інтерфейс програми. Користувач бачить обчислювальні функції, а також поля вводу, куди необхідно ввести параметри генетичного алгоритму - кількість ітерацій алгоритму, розмір популяції та вірогідність мутації. Також за допомогою радіо-кнопок користувач повинен обрати необхідний алгоритм багатокритеріальної оптимізації.

Ідентифікатори елементів інтерфейсу, призначених для вводу початкових даних для роботи програми, та змінні, які вони зберігають, наведені у таблиці 3.1.

Таблиця 3.1 – Зчитувана інформація

|  |  |
| --- | --- |
| Поле | Опис |
| populationSize | Містить значення розміру популяції, яка використовується генетичним алгоритмом. Розмір популяції за замовчуванням - 100 |

Закінчення таблиці 3.1

|  |  |
| --- | --- |
| Поле | Опис |
| iterationLimit | Містить значення кількості ітерацій, після виконання яких генетичний алгоритм завершує свою роботу. Ліміт ітерацій за замовчуванням - сто кроків |
| mutationProbability | Містить значення вірогідності мутації, що застосовується у генетичному операторі мутації для кожного за представлених алгоритмів |
| algo1 | Перша радіо-кнопка, при обиранні якої програма здійснює багатокритеріальну генетичну оптимізацію за допомогою алгоритму RWGA. Обрана за замовченням. |
| algo2 | Друга радіо-кнопка, при обиранні якої програма здійснює багатокритеріальну генетичну оптимізацію за допомогою алгоритму VEGA. |
| algo3 | Третя радіо-кнопка, при обиранні якої програма здійснює багатокритеріальну генетичну оптимізацію за допомогою алгоритму MOGA. |
| algo4 | Четверта радіо-кнопка, при обиранні якої програма здійснює багатокритеріальну генетичну оптимізацію за допомогою алгоритму NPGA. |

Після того, як користувач запустить обчислення, натиснувши на кнопку "Оптимізувати", буде викликана функція Optimize(). Програма за допомогою методу getElementById() зчитає введені користувачем початкові дані.

Залежно від обраного методу оптимізації програма виконає багатокритеріальну оптимізацію за алгоритмами RWGA, VEGA, MOGA або NPGA, кожен з яких описаний у відповідному файлі, визиваючи необхідну функцію - відповідно rwga(), vega(), moga() або npga().

Функція rwga(), що реалізує алгоритм RWGA, знаходиться у файлі rwga.js. У цьому методі оптимізції загальна цільова функція будується з окремих цільових функцій у вигляді їх суми, де для кожної цільової функції визначається окрема вага. 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.

Завантаживши введені користувачем початкові дані, алгоритм створює початкову популяцію індивідів. Розмір популяції зберігається у змінній length, значення якої береться зі введених користувачем параметрів алгоритму. Популяція є набором масивів: inds[], indv[], indl[], indd[], indd0[], де кожний елемент масиву зберігає значення змінних для кожного індивідууму популяції (фенотип особини).

Далі програма ініціює цикл, кількість ітерацій у якому дорівнює змінній limit, яка була задана користувачем.

Потім визначаються випадкові ваги для кожного з критеріїв оптимізації, сума яких дорівнює одиниці.

Для кожної особини додаток вираховує загальну цільову функцію яка складається з кожної цільової функції *fi(x)* за формулою, зазначеною у 3.1.1.

Для селекції найкращих особин програма використовує турнірний відбір, без з пари випадково обраних індивідуумів обирається той, у якого менше значення фітнес-функції.

Далі для отримання нових особин популяції, застосовується механізм лінійного схрещування.

Значення кожної змінної для кожної особини-нащадка знаходиться за допомогою лінійного кросинговеру, де використовуються значення змінних першої та другої батьківської особини, отриманих за допомогою турнірного відбору.

Кожна пара батьківських особин дає два нащадка. Провівши процедуру схрещування *N* разів, отримуємо популяцію нащадків.

На наступному кроці виконується генетичний оператор мутації. З деякою зазначеною вірогідністю значення хромосом особини змінюється на деяке невелике випадкове число. Мутація допомагає підтримувати різноманітність популяції.

Після використання генетичних операторів, батьківська популяція змінюється на популяцію нащадків.

Алгоритм завершує свою роботу тоді, коли буде вичерпано ліміт ітерацій limit у циклу.

Функція vega(), що знаходиться у файлі vega.js, реалізує генетичний алгоритм VEGA. Селекція проводиться за придатністю особин окремо для кожного з критеріїв. Проміжна популяція заповнюється рівними порціями індивідів, відібраних по кожному з окремих критеріїв - для кожного з критеріїв створюється підпопуляція розміром *N / K*, де *N* - розмір всієї популяції. Відбір особин відбувається за допомогою турнірної селекції щодо придатності за кожним критерієм окремо.

Отримавши зчитані з полів вводу початкові дані, алгоритм створює початкову популяцію індивідів. Розмір популяції зберігається у змінній length, значення якої береться зі введених користувачем параметрів алгоритму. Популяція є набором масивів: inds[], indv[], indl[], indd[], indd0[], де кожний елемент масиву зберігає значення змінних для кожного індивідууму популяції.

Далі програма ініціює цикл, кількість ітерацій у якому дорівнює змінній limit, яка була задана користувачем. Для кожного з *K* критеріїв створюється підпопуляція розміром *N/K*, де *N* – розмір всієї популяції. Для кожної підпопуляції вираховується окреме значення фітнес-функції *Fk(x)* по цільовій функції *fk(x).*

Турнірний відбір виконується окремо для кожної з підпопуляцій - обидві особини для порівняння на кожному кроці турніру беруться з однієї підпопуляції.

Для отримання нових особин популяції, застосовується процедури схрещування та мутації. Після використання генетичних операторів, батьківська популяція змінюється на популяцію нащадків.

Алгоритм завершує свою роботу тоді, коли буде вичерпано ліміт ітерацій limit у циклу.

Функція moga(), що знаходиться файлі moga.js, реалізує алгоритм МОGA. Цей алгоритм використовує процедуру ранжирування індивідів, де ранг кожного індивіда визначається числом домінуючих його індивідів, яка називається рангами Голдберга.

Зчитавши з полів вводу початкові дані, алгоритм створює початкову популяцію індивідів. Розмір популяції зберігається у змінній length. Популяція є набором масивів: inds[], indv[], indl[], indd[], indd0[], де кожний елемент масиву зберігає значення змінних для кожної особини.

Далі для кожного індивідууму у популяції визначаються його ранги:

1. з популяції обираються два індивіди - *i* та *j*;
2. для обраних індивідів порівняти один з одним значення фітнес-функцій по кожній функції *fk(x)*;
3. якщо індивід *i* кращий за індивід *j* хоч би по одному з критеріїв оптимізації, та не гірший за рештою критеріїв, тоді *j* - домінуєме рішення відносно *i*;
4. відповідно, якщо індивід *j* кращий за індивід *i* хоч би по одному з критеріїв оптимізації, але не гірший за рештою критеріїв, тоді *i* - домінуєме рішення відносно *j*;
5. ранг домінованої особини збільшується на одиницю.

Таким чином, кожній особині буде присвоєний ранг, що дорівнює кількості особин, якими домінується цей індивід.

Далі програма проводить селекцію найкращих особин за допомогою турнірного відбору. Пари індивідів порівнюються не значення їх фітнес-функцій, а отримані на попередньому кроці ранги.

Для отримання нових особин популяції, застосовуються генетичні оператори схрещування та мутації. Після використання генетичних операторів, батьківська популяція змінюється на популяцію нащадків.

Коли буде вичерпано ліміт ітерацій limit у циклу, алгоритм завершить свою роботу.

У файлі npga.js знаходиться функція npga(), у якій був реалізований генетичний алгоритм багатокритеріальної оптимізації NPGA. У цьому генетичному алгоритмі закладено механізм підтримки різноманітності, що запобігає передчасному сходженню рішення.

Отримавши вхідні дані, алгоритм створює початкову популяцію індивідів. Розмір популяції зберігається у змінній length, значення якої береться зі введених користувачем параметрів алгоритму. Популяція є набором масивів: inds[], indv[], indl[], indd[], indd0[], де кожний елемент масиву зберігає значення змінних для кожного індивідууму популяції.

Потім ініціюється цикл, кількість ітерацій у якому дорівнює змінній limit, яка була зчитана з відповідного поля вводу.

Далі алгоритм використовує оригінальний механізм відбору індивідів, що забезпечує різноманітність популяції:

1. з популяції розміром *N* обирається випадкова порівняльна множина розміром *M*;
2. з популяції обираються два випадкових індивіда *i* та *j*;
3. якщо індивід *i* хоч би за одною цільовою функцією кращий за усі індивіди порівняльної множини, та не гірший за усіма іншими цільовими функціями, тоді індивід *i* - недомінуємий порівняльною множиною;
4. відповідно, якщо індивід *j* хоч би за одною цільовою функцією кращий за усі індивіди порівняльної множини, та не гірший за усіма іншими цільовими функціями, тоді індивід *j* - недомінуємий порівняльною множиною;
5. якщо індивід *i* - недомінуємий , а індивід *j* - домінуємий, тоді індивід *i* перемагає турнір і допускається до схрещування;
6. інакше, коли індивід *j* - недомінуємий , а індивід *i* - домінуємий, тоді індивід *j* перемагає турнір.

Якщо виявилось так, що жодна з цих умов не була виконана, і обидві особини виявилися домінованими чи недомінованими порівняльною множиною, тоді переможцем турніру обирається та особина, у радіусі певної ніші *σshare* якої знаходиться менша кількість сусідніх особин. Це дозволяє підтримувати різноманітність популяції, та запобігати накопичуванню рішень у одній області.

Для отримання нових особин популяції, застосовується процедури схрещування та мутації. Після використання генетичних операторів, батьківська популяція змінюється на популяцію нащадків.

Алгоритм завершує свою роботу тоді, коли буде вичерпано ліміт ітерацій limit у циклу.

Після завершення роботи будь-якого з обраних генетичних алгоритмів, у файлі index.html програма виведе на екран побудовані графіки та результати обчислювання, тобто мінімальні значення цільових функцій, та відповідні значення змінних.

1. АНАЛІЗ РОБОТИ ГЕНЕТИЧНИХ АЛГОРИТМІВ НА ТЕСТОВИХ ПРИКЛАДАХ

4.1 **Перевірка коректності роботи генетичних операторів**

Перед тим, як перевіряти роботу генетичних алгоритмів багатокритеріальної оптимізації, необхідно перевірити роботу загального генетичного алгоритму та усіх генетичних операторів, що приймають участь у його роботі.

Основні етапи роботи класичного генетичного алгоритму такі:

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.

Для перевірки роботи загального генетичного алгоритму були використані функції для яких відоме точне значення мінімуму.

Перша функція, за якою перевірялася робота генетичного алгоритму - це сферична модель, яка вираховується за формулою

Ця функція вважаться досить простою для оптимізації, та має мінімум, що дорівнює 0 у точці *xi* = (0, 0).

Графік функції наведений на рисунку 4.1.

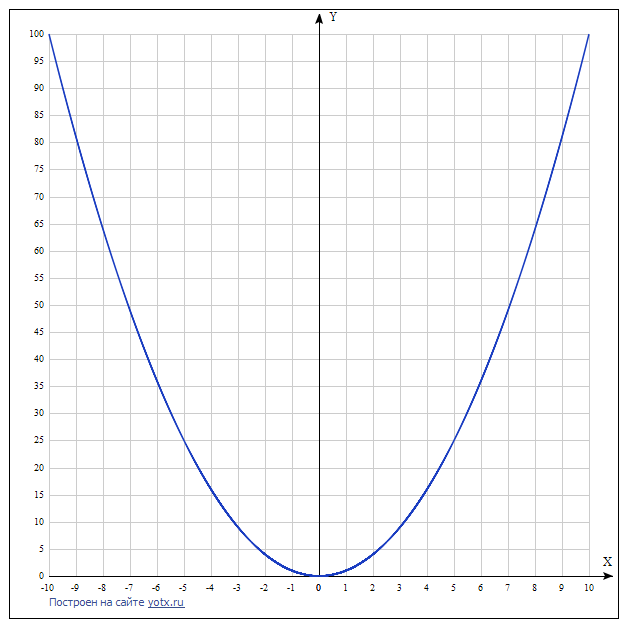


Рисунок 4.1 – Сферична модель для однієї змінної

Перевірка здійснювалася для популяції розміром в 100 особин, з вірогідністю мутації 0.5 та умовою завершення роботи алгоритму після 100 ітерацій. Результати експерименту наведені у таблиці 4.1.

Таблиця 4.1 – Результати роботи алгоритму за першою функцією

|  |  |  |  |
| --- | --- | --- | --- |
| Номер експерименту | Значення цільової функції | Точка мінімуму | Ітерація, на якій був знайдений мінімум |
| 1 | 0 | (0,0) | 14 |
| 2 | 0 | (0,0) | 42 |
| 3 | 0 | (0,0) | 25 |
| 4 | 0 | (0,0) | 22 |
| 5 | 0 | (0,0) | 23 |
| 6 | 0 | (0,0) | 33 |
| 7 | 0 | (0,0) | 14 |
| 8 | 0 | (0,0) | 47 |
| 9 | 0 | (0,0) | 55 |
| 10 | 0 | (0,0) | 16 |

Графік десятого експерименту наведений на рисунку 4.2.



Рисунок 4.2 – Графік значення цільової функції

Як видно, обраних початкових даних було достатньо, щоб на відповідній ітерації знайти точне значення мінімуму перевірочної функції. Результати кожного експерименту співпали з реальним значенням мінімуму функції, отже можна вважати, що перевірка генетичного алгоритму за допомогою сферичної функції завершилася успішно, та усі генетичні оператори працюють коректно.

Друга використана для перевірки алгоритму функція - це функція Растрігіна, яка будується за формулою

Ця функція має складний рельєф, тому вважається досить складною для оптимізації за допомогою генетичних алгоритмів. Мінімальне значення функції дорівнює 0 у точці (0,0).

Графік функції Растрігіна наведений на рисунку 4.3.

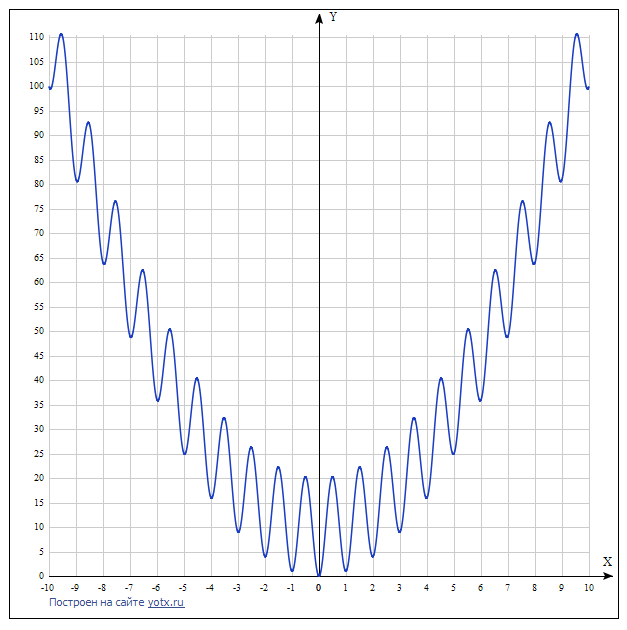


Рисунок 4.3 – Функція Растрігіна

Перевірка роботи алгоритму за допомогою функції Растрігіна здійснювалася для популяції розміром в 100 особин, з вірогідністю мутації 0.5 та умовою завершення роботи алгоритму після 100 ітерацій. Результати експерименту наведені у таблиці 4.2.

Таблиця 4.2 – Результати роботи алгоритму за другою функцією

|  |  |  |  |
| --- | --- | --- | --- |
| Номер експерименту | Значення цільової функції | Точка мінімуму | Ітерація, на якій був знайдений мінімум |
| 1 | 0 | (0,0) | 27 |
| 2 | 0 | (0,0) | 21 |
| 3 | 0 | (0,0) | 41 |
| 4 | 0 | (0,0) | 42 |
| 5 | 0 | (0,0) | 18 |
| 6 | 0 | (0,0) | 36 |
| 7 | 0 | (0,0) | 61 |
| 8 | 0 | (0,0) | 25 |
| 9 | 0 | (0,0) | 34 |
| 10 | 0 | (0,0) | 18 |

Графік десятого експерименту наведений на рисунку 4.4.



Рисунок 4.4 – Графік значення цільової функції

Як показали результати перевірки, обраних початкових даних достатньо, щоб на відповідній ітерації знайти точне значення мінімуму перевірочної функції. Результати кожного експерименту співпали з реальним значенням мінімуму функції Растрігіна, отже можна вважати, що перевірка генетичного алгоритму завершилася успішно.

Отже, був проведений аналіз коректності роботи генетичного алгоритму за допомогою сферичної моделі та функції Растрігіна. Результати перевірки показали, що загальний генетичний алгоритм та усі застосовані у ньому генетичні оператори буди побудовані коректно.

4.2 **Перевірка коректності побудови багатокритеріальних генетичних алгоритмів**

Завершивши перевірку генетичних операторів та загального генетичного алгоритму, переходимо до перевірки генетичних алгоритмів багатокритеріальної оптимізації.

У програмі були використані чотири методи багатокритеріальної оптимізації генетичними алгоритмами, що реалізують різні схеми призначення придатності та селекції:

1. RWGA – Random Weights Genetic Algorithm;
2. VEGA – Vector Evaluated Genetic Algorithm;
3. MOGA – Multiobjective Genetic Algorithm;
4. NPGA – Niched Pareto Genetic Algorithm.

Для перевірки роботи генетичних алгоритмів багатокритеріальної оптимізації були знайдені дві функції з відомими значеннями мінімуму.

Перша функція

,

де значення мінімуму на області [1; 10] по *x* та [1; 10] по *y* дорівнює 0.01 у точці (10, 10).

Друга функція

де значення мінімуму на області [1; 10] по *x* та [1; 10] по *y* дорівнює 0.5013 у точці (10, 10).

Спочатку була проведена перевірка генетичного алгоритму RWGA, результати якої зображено у таблиці 4.3.

Таблиця 4.3 – Результати перевірки методу RWGA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер експерименту | Значення першої цільової функції | Значення другої цільової функції | Точка мінімуму | Ітерація, на якій був знайдений результат |
| 1 | 0.01 | 0.5013 | (10,10) | 15 |
| 2 | 0.01 | 0.5013 | (10,10) | 8 |
| 3 | 0.01 | 0.5013 | (10,10) | 10 |
| 4 | 0.01 | 0.5013 | (10,10) | 8 |
| 5 | 0.01 | 0.5013 | (10,10) | 8 |
| 6 | 0.01 | 0.5013 | (10,10) | 13 |
| 7 | 0.01 | 0.5013 | (10,10) | 12 |
| 8 | 0.01 | 0.5013 | (10,10) | 7 |
| 9 | 0.01 | 0.5013 | (10,10) | 16 |
| 10 | 0.01 | 0.5013 | (10,10) | 12 |

Результати перевірки співпадають з відомими значеннями мінімумів функцій та точок мінімуму тобто метод RWGA працює коректно.

Далі був перевірений метод VEGA, результати перевірки наведені у таблиці 4.4.

Таблиця 4.4 – Результати перевірки методу VEGA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер експерименту | Значення першої цільової функції | Значення другої цільової функції | Точка мінімуму | Ітерація, на якій був знайдений результат |
| 1 | 0.01 | 0.5013 | (10,10) | 6 |
| 2 | 0.01 | 0.5013 | (10,10) | 8 |
| 3 | 0.01 | 0.5013 | (10,10) | 8 |
| 4 | 0.01 | 0.5013 | (10,10) | 16 |
| 5 | 0.01 | 0.5013 | (10,10) | 9 |
| 6 | 0.01 | 0.5013 | (10,10) | 10 |
| 7 | 0.01 | 0.5013 | (10,10) | 13 |
| 8 | 0.01 | 0.5013 | (10,10) | 8 |
| 9 | 0.01 | 0.5013 | (10,10) | 11 |
| 10 | 0.01 | 0.5013 | (10,10) | 7 |

Результати усіх проведених експериментів співпадають з відомими значеннями мінімумів функцій та точок мінімуму, отже можна вважати, що генетичний алгоритм VEGA побудований коректно.

Наступний перевірений метод - генетичний алгоритм MOGA, результати експериментів наведені у таблиці 4.5.

Таблиця 4.5 – Результати перевірки методу MOGA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер експерименту | Значення першої цільової функції | Значення другої цільової функції | Точка мінімуму | Ітерація, на якій був знайдений результат |
| 1 | 0.01 | 0.5013 | (10,10) | 6 |
| 2 | 0.01 | 0.5013 | (10,10) | 16 |
| 3 | 0.01 | 0.5013 | (10,10) | 11 |
| 4 | 0.01 | 0.5013 | (10,10) | 11 |
| 5 | 0.01 | 0.5013 | (10,10) | 8 |
| 6 | 0.01 | 0.5013 | (10,10) | 7 |
| 7 | 0.01 | 0.5013 | (10,10) | 9 |
| 8 | 0.01 | 0.5013 | (10,10) | 10 |
| 9 | 0.01 | 0.5013 | (10,10) | 4 |
| 10 | 0.01 | 0.5013 | (10,10) | 6 |

Як бачимо, результати перевірки співпадають з відомими значеннями мінімумів функцій та точок мінімуму, отже генетичний алгоритм MOGA працює коректно.

Останній з методів, що був використаний у додатку, був метод NPGA. Результати його роботи наведені у таблиці 4.6.

Таблиця 4.6 – Результати перевірки методу NPGA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер експерименту | Значення першої цільової функції | Значення другої цільової функції | Точка мінімуму | Ітерація, на якій був знайдений результат |
| 1 | 0.01 | 0.5013 | (10,10) | 10 |
| 2 | 0.01 | 0.5013 | (10,10) | 7 |
| 3 | 0.01 | 0.5013 | (10,10) | 16 |
| 4 | 0.01 | 0.5013 | (10,10) | 12 |
| 5 | 0.01 | 0.5013 | (10,10) | 13 |
| 6 | 0.01 | 0.5013 | (10,10) | 19 |
| 7 | 0.01 | 0.5013 | (10,10) | 14 |
| 8 | 0.01 | 0.5013 | (10,10) | 11 |
| 9 | 0.01 | 0.5013 | (10,10) | 16 |
| 10 | 0.01 | 0.5013 | (10,10) | 8 |

У результати перевірки виявилося, що значення, вирахувані алгоритмом, співпадають з відомими значеннями мінімумів функцій та точок мінімуму. Отже генетичний алгоритм NPGA побудований коректно.

Як виявилося у результаті проведеної перевірки, усі чотири використані у додатку генетичні алгоритми багатокритеріальної оптимізації були побудовані коректно.

4.3 **Пошук оптимальних початкових даних для багатокритеріальних генетичних алгоритмів**

Для коректної роботи генетичного алгоритму дуже важливо знайти значення розміру популяції, кількості ітерацій та вірогідності мутації, що забезпечують правильність результатів. Для пошуку необхідних значень цих початкових даних був проведений аналіз.

Зафіксуємо розмір популяції у 4 особини, а вірогідність мутації будемо змінювати як 0.1, 0.3 та 0.5. Результати експерименту наведені у таблиці 4.7.

Таблиця 4.7 – Результати експерименту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кількість ітерацій | Значення цільових функцій на останньому кроці алгоритму | | | |
| RWGA | VEGA | MOGA | NPGA |
| Вірогідність мутації 0.1 | | | | |
| 10 | 07915  0,0239 | 0,7577  0,0241 | 0,5696  0,0129 | 0,7632  0,231 |
| 15 | 0,7210  0,0201 | 0,6875  0,1865 | 0,7796  0,0241 | 0,8247  0,0270 |
| 25 | 0,5728  0,0130 | 0,5582  0,0133 | 0,0543  0,0117 | 0,899  0,0322 |
| 50 | 0,5016  0,0100 | 0,6717  0,0192 | 0,5052  0,0101 | 0,5014  0,0100 |
| 100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5014  0,0100 |
| Вірогідність мутації 0.3 | | | | |
| 10 | 0,6235  0,0157 | 0,6100  0,0142 | 0,5014  0,0100 | 0, 9535  0,0364 |
| 15 | 0,5487  0,0117 | 0,6149  0,0189 | 0,5568  0,0123 | 0,7028  0,0196 |
| 25 | 0,6589  0,1696 | 0,5786  0,0138 | 0,5098  0,0103 | 0,5104  0,0103 |
| 50 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 |
| 100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 |
| Вірогідність мутації 0.5 | | | | |
| 10 | 0,6661  0,0171 | 0,5466  0,0116 | 0,6112  0,0149 | 0,6652  0,0176 |
| 15 | 0,5826  0,0149 | 0,6226  0,0160 | 0,5065  0,0102 | 0,6040  0,0145 |
| 25 | 0,5019  0,0100 | 0,5013  0,0100 | 0,5214  0,0108 | 0,5912  0,0139 |
| 50 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 |
| 100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 |

Зафіксуємо розмір популяції у 8 особин, а вірогідність мутації будемо змінювати як 0.1, 0.3 та 0.5. Результати експерименту наведені у таблиці 4.8.

Таблиця 4.8 – Результати експерименту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кількість ітерацій | Значення цільових функцій на останньому кроці алгоритму | | | |
| RWGA | VEGA | MOGA | NPGA |
| Вірогідність мутації 0.1 | | | | |
| 10 | 0,6002  0,0146 | 0,6387  0,0162 | 0,7652  0,0232 | 0,6002  0,0144 |
| 15 | 0,6269  0,0154 | 0,5526  0,0124 | 0,6381  0,0162 | 0,6554  0,0170 |
| 25 | 0,5261  0,0105 | 0,5236  0,0107 | 0,5014  0,0100 | 0,5781  0,0132 |
| 50 | 0,5014  0,0100 | 0,5020  0,0100 | 0,5025  0,0100 | 0,5295  0,0111 |
| 100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 |
| Вірогідність мутації 0.3 | | | | |
| 10 | 0,5113  0,0105 | 0,7006  0,0184 | 0,5328  0,0112 | 0,7738  0,0238 |
| 15 | 0,5536  0,0125 | 0,5508  0,0127 | 0,5218  0,0108 | 0,7047  0,0197 |
| 25 | 0,5148  0,0100 | 0,5102  0,0101 | 0,5013  0,0100 | 0,5354  0,0114 |
| 50 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 |
| 100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 |
| Вірогідність мутації 0.5 | | | | |
| 10 | 0,5891  0,0125 | 0,5299  0,113 | 0,6014  0,0143 | 0,6286  0,0157 |
| 15 | 0,5268  0,0111 | 0,5597  0,0127 | 0,5013  0,0100 | 0,5498  0,0120 |
| 25 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 |
| 50 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 |
| 100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 | 0,5013  0,0100 |

За результатами перевірки маємо, що при розмірі популяції у 4 особини, вірогідності мутації менше за 0.5 та кількості ітерацій менше за 50 можуть виникнути розбіжності у результатах. При розмірі популяції у 8 особин, вірогідності мутації у 0.5 та кількості ітерацій більш ніж 50, значення цільових функцій для усіх генетичних алгоритмів співпадають при багатьох перевірках. Ці значення можна вважати знайденими мінімумами функцій, які дорівнюють *min f1* = 0,5013 та *min f2* = 0,01 у точці *x* = 10, *y* = 10.

Приклад побудованого програмою графіку значень цільових функцій наведено на рисунку 4.5.



Рисунок 4.5 – Графік значень цільових функцій за методом MOGA

1. РОЗВ’ЯЗАННЯ ЗАДАЧІ ТОКАРНОЇ ОБРОБКИ ТИТАНУ

У попередньому розділі роботи завдяки моделюванню тестових функцій було встановлено, що алгоритм обчислювання складений вірно, а розроблений додаток працює коректно. Також був проведений пошук оптимальних початкових показників генетичного алгоритму для використаних прикладів. Тепер перейдемо до рішення задачі токарної обробки титану з субмікро- і нанокристалічною структурою за допомогою розробленого додатку, де мінімізуються функція розміру зерна металу *f1* та функція ефективності *f2*.

Зафіксуємо розмір популяції у чотири особини, а вірогідність мутації будемо змінювати як 0.1, 0.3 та 0.5. Результати експерименту наведені у таблиці 5.1.

Таблиця 5.1 – Результати експерименту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кількість ітерацій | Значення цільових функцій на останньому кроці алгоритму | | | |
| RWGA | VEGA | MOGA | NPGA |
| Вірогідність мутації 0.1 | | | | |
| 10 | 2,0127  0,000000083 | 4,5687  0,000000167 | 1,0792  0,000000065 | 1,1356  0,000000167 |
| 15 | 1,9755  0,000000169 | 5,4469  0,000000145 | 1,951  0,000000197 | 1,834  0,000000094 |
| 25 | 1,824  0,000000094 | 1,706  0,000000188 | 1,7866  0,000000068 | 2,0736  0,000000081 |
| 50 | 1,002  0,000000081 | 1,0564  0,000000085 | 0,977517  0,000000042 | 0,9856  0,000000116 |
| 100 | 0,9801  0,000000027 | 0,9791  0,000000027 | 0,9775  0,000000026 | 0,9775  0,000000033 |
| Вірогідність мутації 0.3 | | | | |
| 10 | 2,9372  0,000000113 | 2,6202  0,000000141 | 1,3849  0,000000099 | 3,1194  0,000000120 |
| 15 | 1,1200  0,000000090 | 1,0808  0,00000079 | 1,3411  0,000000096 | 0,9775  0,000000051 |
| 25 | 1,085  0,000000057 | 1,1794  0,000000094 | 0,9881  0,000000026 | 0,9775  0,00000045 |

Завершення таблиці 5.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кількість ітерацій | RWGA | VEGA | MOGA | NPGA |
| 50 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9813  0,000000026 |
| 100 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 |
| Вірогідність мутації 0.5 | | | | |
| 10 | 1,9351 | 2,2609  0,000000080 | 1,8923  0,000000097 | 1,1608  0,000000059 |
| 15 | 0,9783 | 0,9779  0,000000071 | 0,9776  0,000000034 | 1,1338  0,000000125 |
| 25 | 0,9775 | 0,9775  0,000000032 | 0,9775  0,000000033 | 1,0120  0,000000026 |
| 50 | 0,9775 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 |
| 100 | 0,9775 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 |

Приклад графіку цільових функцій, що будує програма під час роботи, наведено на рисунку 5.1, де проводиться оптимізація за методом NPGA для

популяції у чотири особини, 50 ітерацій та вірогідності мутації 0,5.



Рисунок 5.1 - Графік значень цільових функцій за методом NPGA

Далі оберемо розмір популяції у 12 особин, а вірогідність мутації будемо змінювати як 0.1, 0.3 та 0.5. Результати експерименту наведені у таблиці 5.2.

Таблиця 5.2 – Результати експерименту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кількість ітерацій | Значення цільових функцій на останньому кроці алгоритму | | | |
| RWGA | VEGA | MOGA | NPGA |
| Вірогідність мутації 0.1 | | | | |
| 10 | 1,4211  0,000000123 | 1,3375  0,000000011 | 1,4360  0,000000100 | 2,407  0,000000131 |
| 15 | 1,2349  0,000000109 | 1,1690  0,000000115 | 1,1995  0,000000107 | 1,6314  0,000000164 |
| 25 | 1,1183  0,000000093 | 1,3812  0,000000060 | 1,1514  0,000000086 | 0,8996  0,000000088 |
| 50 | 0,9972  0,000000039 | 1,1001  0,000000044 | 0,9939  0,000000102 | 1,2427  0,000000040 |
| 100 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000044 |
| Вірогідність мутації 0.3 | | | | |
| 10 | 1,8531  0,000000054 | 1,5345  0,000000033 | 1,9592  0,000000086 | 1,7940  0,000000117 |
| 15 | 1,2117  0,000000042 | 1,0075  0,000000084 | 1,509  0,000000035 | 0,9775  0,000000059 |
| 25 | 0,9789  0,000000028 | 0,9787  0,000000036 | 0,9775  0,000000029 | 0,9775  0,000000044 |
| 50 | 0,9775  0,000000027 | 0,9775  0,000000027 | 0,9775  0,000000026 | 0,9775  0,000000027 |
| 100 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000027 |
| Вірогідність мутації 0.5 | | | | |
| 10 | 1,0918  0,000000030 | 1,0296  0,000000055 | 1,0435  0,000000070 | 1,1523  0,000000069 |
| 15 | 0,9775  0,000000041 | 1,0082  0,000000035 | 0,9775  0,000000028 | 0,9775  0,000000030 |
| 25 | 0,9775  0,000000028 | 0,9775  0,000000030 | 0,9775  0,000000026 | 0,9775  0,000000027 |
| 50 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 |
| 100 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 | 0,9775  0,000000026 |

Отже, після проведеного аналізу результатів оптимізації маємо, що при розмірі популяції у 4 особини, вірогідності мутації менше за 0.5 та кількості ітерацій менше за 50 можуть виникнути розбіжності у результатах. При розмірі популяції у 12 особин, вірогідності мутації у 0.5 та кількості ітерацій біль ніж 50, значення цільових функцій для усіх генетичних алгоритмів співпадають при багатьох перевірках. Ці значення можна вважати знайденими мінімумами функцій, які дорівнюють *min f1* = 0,9775 та *min f2* = 0,000000026 у точці *s* = 0.0123, *v* = 100, *L* = 0.1, *d* = 0.01 *D0* = 2e-8.

Інтерфейс додатку з результатами роботи програми та побудованими графіками зображено на рисунку 5.2.



Рисунок 5.2 - Результати оптимізації

Далі проведемо порівняльний аналіз роботи різноманітних генетичних алгоритмів за швидкістю пошуку оптимального значення.

Для цього проведемо серію експериментів, встановивши розмір популяції у 12 особин, кількість ітерацій у 50 та вірогідність мутації у 0.5.

Результати наведені у таблиці 5.3.

Таблиця 5.3 - Порівняльний аналіз генетичних алгоритмів

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер експерименту | Ітерація, на якій був знайдений результат | | | |
| RWGA | VEGA | MOGA | NPGA |
| 1 | 18 | 16 | 12 | 20 |
| 2 | 16 | 22 | 18 | 37 |
| 3 | 15 | 24 | 21 | 22 |
| 4 | 26 | 25 | 15 | 24 |
| 5 | 19 | 16 | 22 | 32 |
| 6 | 23 | 18 | 16 | 20 |
| 7 | 18 | 24 | 23 | 18 |
| 8 | 19 | 21 | 21 | 22 |
| 9 | 20 | 22 | 13 | 19 |
| 10 | 19 | 20 | 14 | 26 |
| Середнє значення | 19,3 | 20,8 | 17,5 | 24 |

Отже, за результатами проведеної перевірки було встановлено, що найшвидше оптимальне рішення знаходить метод MOGA, що доводить високу продуктивність генетичних алгоритмів, заснованих на використанні рангів Парето. В той час найповільніше рішення знаходить метод NPGA, бо його механізм збереження різноманітності популяції та запобігання передчасного сходження алгоритму сповільнює процес оптимізації.

6 ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ

6.1 **Актуальність теми**

Теорія багатокритеріальної оптимізації з’явилася понад півстоліття тому. Завдання в цій області природним чином виникли в математичній економіці, а надалі розроблялися підходи до вирішення даного роду завдань у різних галузях: фахівцями з системного аналізу і теоретиками в області прийняття рішень. Як показує багаторічний досвід фахівців в області вирішення оптимізаційних задач, застосування традиційних методів оптимізації не завжди дозволяє досягти бажаного результату - дійсної точки оптимуму за прийнятний час, або, принаймні, для цього потрібні значні обчислювальні ресурси. Тому має сенс розробляти нові напрямки в області вирішення складних завдань оптимізації, які б дозволили уникнути основних недоліків класичних методів. До таких нових напрямків, наприклад, можна віднести еволюційний підхід, який, незважаючи на свій досить молодий вік, вже встиг зарекомендувати себе як дуже ефективний метод вирішення широкого класу задач.

У даній дипломній роботі розглядаються особливості побудови та реалізації програмного забезпечення для вирішення багатокритеріальних задач за допомогою генетичних алгоритмів. Розроблене програмне забезпечення призначене для вирішення задачі оптимізації режимів обробки титану з субмікро- та нанокристалічною структурою.

6.2 **Огляд існуючих методів розв'язку поставленого задачі**

Метою дипломної роботи була розробка програмного забезпечення для вирішення задач багатокритеріальної оптимізації за допомогою генетичних алгоритмів.

Більшість розв'язуваних практичних завдань передбачають пошук рішення, що є оптимальним згідно з кількома критеріями. Однак, більшість методів, використовуваних для вирішення цих завдань, використовують єдиний, складовий оптимізаційний критерій. У цьому випадку завдання багатокритеріальної оптимізації зводиться до однієї або декількох задач однокритеріальної оптимізації. Існує величезна різниця між двома цими завданнями. При однокритеріальній оптимізації здійснюється пошук єдиного оптимального рішення. При багатокритеріальної оптимізації здійснюється пошук декількох оптимальних рішень, що дозволяє рівним чином враховувати всі критерії, що оптимізуються. Після завершення оптимізації користувач має можливість вибрати найкраще з його точки зору рішення, яке представляє собою компроміс між декількома суперечливими критеріями. Пошук безлічі рішень при багатокритеріальної оптимізації ґрунтується на концепції Парето-оптимальності.

Генетичні алгоритми добре зарекомендували себе в якості методик пошуку у багатьох областях практично при повній відсутності інформації про властивості цільової функцій і обмежень. В різних дослідженнях було розроблено декілька методів і підходів використання генетичних алгоритмів для вирішення багатокритеріальної оптимізації.

Програма призначена для багатокритеріальної оптимізації режимів обробки титану з субмікро- та нанокристалічною структурою, де оптимізуються функцій продуктивності та розміру зерна металу за допомогою генетичного алгоритму. Це дозволяє підвищити ефективність виробництва, обробивши заготовку з максимальною швидкістю за найменший час, але водночас не дає титану втратити властивості субмікро- та нанокристалічної структури від занадто швидкої обробки.

6.3 **Розрахунок кошторису витрат на НДР**

У ринковій економіці інтелектуальна власність може продаватися, тому треба знати за якою ціною можна це зробити. У зв'язку з цим, далі представлено розрахунок вартості такого договору куплі продажу об'єктів інтелектуальної власності.

Виконання наукових досліджень вимагає визначених витрат, які необхідно розглядати як додаткові капіталовкладення або інвестиції. Витрати на проведення НДР відносяться до перед виробничих витрат. Це одноразові витрати, що включать у себе всі роботи, які виконуються при проведенні роботи.

Кошторис включає витрати та прибуток розробника. Витрати включають:

- матеріальні витрати;

- витрати на оплату праці;

- відрахування на соціальні заходи;

- амортизаційні відрахування;

- інші операційні витрати.

Плановий кошторис витрат складається за укрупненими статями витрат.

6.3.1 **Витрати на оплату праці**

До витрат на оплату праці відносяться заробітна плата персоналу, зайнятого безпосередньо при виконанні конкретної роботи.

Розрахунок основної заробітної плати виконавців конкретної теми НДР (дипломної роботи) виконується на основі штатного розкладу і планової кількості виконавців, і наведений у таблиці 6.1.

Таблиця 6.1 – Витрати на оплату праці

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Посада | Оклад на місяць, грн. | Кількість виконавців | Кількість місяців роботи за рік | Сума, грн. |
| Керівник проекту | 4500,00 | 1 | 1 | 4500,00 |
| Програміст | 3000,00 | 1 | 5 | 15000,00 |
| Разом | | | | 19500,00 |

6.3.2 **Відрахування до бюджету**

Відрахування до бюджету на соціальні заходи становлять 36% фонду оплати праці:

19500,000.36=7020.

6.3.3 **Витрати на матеріали**

Витрати на матеріали, канцелярсько-письмові приналежності розраховуються за їхньою кількістю й прейскурантними цінами. Матеріальні витрати наведені в таблиці 6.2.

Таблиця 6.2 – Матеріальні витрати

|  |  |  |  |
| --- | --- | --- | --- |
| Найменування | Вартість, грн. | Кількість, шт. | Сума, грн. |
| Лазерний диск (DVD-RW) | 10,00 | 1 | 10,00 |
| Папір А4 (пачка папера) | 65,00 | 1 | 65,00 |
| Папір А1 | 10,00 | 3 | 30,00 |
| Роздруківка на принтері | 0,45 | 120 | 54,00 |
| Папка для проекту | 20,00 | 1 | 20,00 |
| Сума | | | 179,00 |

6.3.4 **Амортизаційні відрахування**

Амортизаційні відрахування обчислювальної техніки розраховуються за формулою:

,

де а – норма амортизації, що складає 25%;

*S* – залишкова вартість ПК;

 – період виконання дипломної роботи.

Наприклад, первісна вартість ПК складає 7500 грн., норма амортизації дорівнює 1875 грн. за рік. Термін використання ПК 2 рік, тоді залишкова вартість – 3750 грн.

При обчисленні амортизаційних відрахувань отримуємо:

Амортизаційні відрахування робочого місця розрахуємо за формулою:

,

де *а* – норма амортизації приміщення, що складає 5%.

Отримуємо амортизаційні відрахування робочого місця:

*S* – вартість оренди приміщення – 4000 грн. за кв.м./місяць (якщо використовується орендоване приміщення). На одну людину відводиться 6 кв.м.

6.3.5 **Витрати на електроенергію**

Розрахунок вартості електроенергії для процесу:

,

де *М* – це споживана потужність, яка дорівнює наприклад, 650 Вт,

*Т тар* – тариф за 1кВт, що дорівнює 1,1818 грн.,

*Т* – кількість годин роботи, що складає 660 годин.

При обчисленні вартості електроенергії для процесу отримуємо суму 0,6\*0,650\*1,1818\*660=304,19 грн.

Розрахунок вартості освітлювальної електроенергії проводимо за тією ж формулою, але за *М* приймаємо споживану потужність штучного освітлення (120 Вт) та *Т* - це кількість годин роботи при штучному освітленні (250 годин). Отримуємо результат – 0,120\*1,1818\*250=35,454 грн.

Сумарні витрати на електроенергію дорівнюють 339,644 грн.

6.3.6 **Сума витрат на НДР**

Разом витрат 27929,27 грн., що становлять 70% кошторису. Усього кошторис витрат на НДР – 39898,95 грн. Прибуток складає 30% і становить 11969,68 грн. Усі розрахунки та результати наведені в таблиці 6.3.

Таблиця 6.3 – Розрахунок кошторису витрат на НДР

|  |  |  |
| --- | --- | --- |
| Найменування статей витрат | Методика розрахунку | Величина, грн. |
| Заробітна плата робітників | Штатний розклад | 19500 |
| Відрахування на соціальні й інші види страхування | 36% заробітної плати | 7020 |
| Амортизаційні відрахування обчислювальної техніки | ,  де *а* – норма амортизації - 25%,  *S* – залишкова вартість ПК | 390,00 |
| Вартість матеріалів | Таблиця 6.2 | 179,00 |
| Амортизаційні відрахування робочого місця | ,  де *а* – норма амортизації приміщення, що складає 5%,  *S* – вартість оренди приміщення | 500,00 |

Закінчення таблиці 6.3

|  |  |  |
| --- | --- | --- |
| Найменування статей витрат | Методика розрахунку | Величина, грн. |
| Вартість електроенергії для процесу | ,  де *М* – це споживана потужність,  *Т тар* – тариф за 1кВт,  *Т* – кількість годин роботи | 304,00 |
| Вартість електроенергії освітлювальної | ,  де *М* – це споживана потужність,  *Т тар* – тариф за 1кВт,  *Т* – кількість годин роботи при штучному висвітленні | 35,00 |
| Разом витрат | 70% | 27928,00 |
| Прибуток | 30% | 11969,00 |
| Усього кошторису витрат на НДР | 100% | 39897,00 |

Усі величини витрат та прибутку округляються до цілої величини, тому що метод розрахунку таблиці 6.3 має похибку. Час розробки НДР – Т2=5 місяців.

6.4 **Розрахунок науково-технічного ефекту**

Ефективність експериментального дослідження та НДР оцінюється на основі групи показників, що характеризують ступінь впливу наукового резуль­тату на різні сторони громадського життя.

Узагальнений кількісний показник науково-технічного ефекту розраховується по формулі:



де ** – вагові коефіцієнти *i*-го показника;

 – оцінки по *i*-му показнику.

Значення  і  вибираються на основі експертних оцінок відповідно до ознак науково-технічного ефекту, наприклад, представлені у таблиці 6.4.

Таблиця 6.4 – Дані для розрахунку науково-технічного ефекту (НТЕ)

|  |  |  |
| --- | --- | --- |
| Ознаки науково-технічного рівня | Значення | Значення ,  бали |
| Перспективність | 0.2 | 8 |
| Можливість реалізації | 0.5 | 7 |
| Новизна | 0.3 | 8 |

Науково-технічний рівень (новизна) - показує наближення до світових досягнень. Характеризується позитивним рішенням поставлених задач на основі простих узагальнень, аналізом зв'язків між фактами, поширенням відомих принципів на нові об'єкти.

Перспективність характеризується тим, що буде сприяти в майбутньому підвищенню суспільної продуктивності праці, задоволенню знову виникаючих потреб [26].

Можливість реалізації. Характеристика ознаки - галузевий масштаб реалізації.

Підставляючи у вхідну формулу відповідні значення показників, отримаємо:

*НТЕ* =0,2\*8+0,5\*7+0,3\*8=7,5.

Порівнюючи отримане значення науково-технічного ефекту НДР з макси­мальним значенням показника науково-технічного ефекту, що узагальнюється, рівним 10 балам, можна зробити висновок, що робота є вища за середній рівень науково-технічного ефекту НДР.

6.5 **Розрахунок економічного ефекту та ефективності виробництва від впровадження НДР**

Проведемо оцінку економічного ефекту, якій очікується.

Економічний ефект, прибуток (Ееф) розрахуємо за формулою:

,

де  - сумарна річна величина доходу, якій очікується отримати у результаті впровадження НДРМ;

 - капіталовкладення, у нашому випадку - сума кошторису витрат на НДРМ і капіталовкладень на впровадження результатів НДРМ.

Як бачимо з формули (4.7), економічний ефект (прибуток) залежить від величини , яка може бути отримана всілякими шляхами. У нашому випадку сумарна річна величина доходу розраховується в такий спосіб:

 ,

де Е1 – дохід від підвищення достовірності результату;

Е2 – дохід від реалізації результатів НДРМ іншим організаціям.

Дохід від підвищення достовірності результату та дохід від реалізації результатів НДРМ іншим організаціям розраховуються за формулами:

,

,

де  - вартість товарів і послуг, кількість яких залежить від підвищення достовірності результату (експертна оцінка);

k1 - достовірність рішення при використанні непрямих методів (експертна оцінка);

k2 - достовірність рішення, при використанні методів, запропонованих у даному дипломному проекті (експертна оцінка).

Втехн - вартість технології, запропонованої в даної НДРМ (експертна оцінка);

N - кількість потенційних покупців даної технології (експертна оцінка).

Як вже було зазначено раніше, у якості експерта виступив керівник дипломного проекту.

Експертні оцінки і рекомендації за значенням показників, які визначають величину економічного ефекту від впровадження НДРМ наведені у таблиці 6.5.

Таблиця 6.5 - Експертні оцінки і рекомендації за значенням показників, які визначають величину економічного ефекту

|  |  |
| --- | --- |
| Показники, які визначають величину економічного ефекту | Значення |
| Вартість товарів і послуг, кількість яких залежить від підвищення достовірності результату, грн. | 15000 |
| Достовірність рішення при використанні непрямих методів | 0,6 |
| Достовірність рішення, при використанні методів, запропонованих у даному дипломному проекті | 0,9 |
| Вартість технології, запропонованої в даної НДРМ, грн. | 5000 |
| Кількість потенційних покупців даної технології | 6 |

Тобто, на підставі оцінки експерту: 15000 грн.; k1=0,6; k2=0,9; Втехн=5000; N =6.

Таким чином, дохід від підвищення достовірності результату:

 15000\*(0,9-0,6)= 4500 грн.

Дохід від реалізації результатів НДРМ іншим організаціям:

 5000 \* 6 = 30000 грн.

Сумарна річна величина доходу, яка очікується у результаті впровадження НДРМ:

 4500 + 30000 = 34500 грн.

Таким чином, очикуємий економічний ефект (прибуток), від впровадження НДРМ складає:

 34500 – 27928 = 6572 грн. за рік.

Економічну ефективність (рентабельність) НДРМ () розрахуємо за формулою:



Отже,  6572 / 27928 \* 100 = 23.5%.

Строк окупності капіталовкладень () розрахуємо за формулою:



Таким чином, 27928 / 34500 = 0,8 років, або 9,6 місяців.

Результати оцінки науково-технічної результативності та очікуваного економічного ефекту НДРМ зведено в таблицю 6.6.

Таблиця 6.6 – Оцінка науково-технічної результативності та очікуваного економічного ефекту НДРМ

|  |  |
| --- | --- |
| Найменування показника | Значення |
| Коефіцієнт науково-технічної результативності НДРМ | 0,61 |
| Економічний ефект (прибуток) від впровадження НДРМ, грн. за рік | 6572 |
| Економічну ефективність НДРМ, % | 23,5%. |
| Строк окупності капіталовкладень, міс | 9,6 |

7 ОХОРОНА ПРАЦІ

7.1 **Загальні питання охорони праці**

Охорона праці є необхідною та важливою складовою кожного робочого процесу, тому що виявляє та впроваджує заходи попередження, усунення причин та знешкодження наслідків нещасних випадків на виробництві, професійних захворювань, аварій пожеж та інше.

Одним з основних завдань охорони праці є забезпечення для людини сприятливих та безпечних умов праці, за яких виключена можливість дії на працівника небезпечних та шкідливих факторів. Основні положення охорони праці містяться в наступних правових та нормативно-технічних документах: Конституції України, Законі України “Про охорону праці”[39], Кодексі законів про працю [40], ССБТ, ССОП та інших.

Тема дипломної роботи: "Розробка програмного забезпечення для розв’язання задач багатокритеріальної оптимізації генетичними алгоритмами", призначена для розробки додатку, що виконує багатокритеріальну оптимізацію цільових функції за допомогою генетичних алгоритмів.

В даному розділі розглянуті питання охорони праці для розробки програмного продукту в робочому класі з використанням персонального комп’ютера, тому питання охорони навколишнього середовища не розглядаються.

7.2 **Небезпечні та шкідливі чинники виробничого приміщення**

Характеристика приміщення, в якому виконувалася дипломна робота, а також в якому знаходиться комп’ютер, розташований в НТУ «ХПІ» на кафедрі «Системи і процеси управління» (СПУ). Площа приміщення становить 10,5 м², воно включає в себе одне робоче місце. Наведені розміри відповідають санітарним нормам, затвердженим НПАОП 0.00-1.28-10 [41], згідно з якими площа повинна бути не менше 6 м² на одне робоче місце.

Робота користувача персонального комп’ютера виконується в одноманітній позі в умовах обмеження загальної м’язової активності при рухливості кистей рук, великій напрузі зорових функцій і нервово-емоційному напруженні під впливом різних фізичних факторів: електростатичного поля; електромагнітних випромінювань в над низько частотному, низькочастотному і середньому діапазонах (5 Гц - 400 кГц); рентгенівського, ультрафіолетового, інфрачервоного випромінювань, випромінювань видимого діапазону, акустичного шуму; незадовільного рівня освітленості, незадовільних метеорологічних умов.

При використанні ЕОМ на людину діє ряд шкідливих і небезпечних виробничих факторів, наведених у таблиці 7.1 за ГОСТ 12.0.003-74 \* [42].

Таблиця 7.1- Перелік небезпечних і шкідливих чинників на робочому місці оператора в приміщені з ЕОМ

|  |  |  |
| --- | --- | --- |
| Найменування чинника | Джерело виникнення | Нормована величина і її значення. |
| 1. Фізичні фактори | | |
| 1 Підвищений рівень шуму | Вентилятор, система освітлення, друкуючий пристрій | Рівень звуку La≤50 дБ (А) |
| 2 Підвищене значення напруги в електричній мережі | Блок живлення | І=0,6mA, U = 36B |
| 3 Рентгенівське випромінювання | Монітор ЕОМ | Dекв =100 мкР/год |

Закінчення таблиці 7.1

|  |  |  |
| --- | --- | --- |
| Найменування чинника | Джерело виникнення | Нормована величина і її значення. |
| 4 Рівень електромагнітних випромінювань | ЭПТ монітора, системний блок, мережа живлення | Відстань-50 см близько ПК 2-5кГц-25В / м |
| 5 Ультрафіолетове випромінювання | Комп'ютер | Щільність потоку ультрафіолетового випромінювання 10Вт/ м |
| 6 Недолік природного освітлення | Неправильне планування розташування комп'ютерів | КПО,eH ≥ 1,5% |
| 7 Вібрація | Вентиляційна система | Віброприскорення, м/с2; віброшвидкість, м/с або їх рівні, дБ; La = 33 дБ, L*v* = 75 дБ |

7.3 **Виробнича санітарія**

7.3.1 **Мікроклімат виробничого приміщення**

Показниками, що характеризують метеорологічні умови в закритих виробничих приміщеннях ( мікроклімат ) є:

- температура повітря , ° С;

- відносна вологість повітря , %;

- швидкість руху повітря , м / с.

При роботах операторського типу, пов’язаних з великим нервово - емоційною напругою, передбачені оптимальні значення параметрів мікроклімату, які поширюються на робочу зону. Роботи виконуються сидячи і належать до категорії 1а - легкі фізичні. Енерговитрати становлять до 139 Вт. Оптимальні норми температури, відносної вологості, швидкості руху повітря в приміщенні і категорія робіт в холодний і теплий період року приведені в таблиці 7.2 .

Таблиця 7.2 - Оптимальні параметри метеорологічних умов на робочому місці в приміщенні з ЕОМ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Період року | Категорія робіт по тяжкості | Температура, °С | Відносна вологість,% | Швидкість руху повітря, Vм/с |
| Холодний | Легка - 1а | 22-24 | 40-60 | 0.1 |
| Теплий | Легка - 1а | 23-25 | 40-60 | 0.1 |

Досягнення нормативних метеорологічних умов приходить за допомогою кондиціонерів, в зимовий час приміщення опалюють. У холодний період року обмін повітря здійснюється за допомогою провітрювання та опалення відповідно до ДБН В.2.5-67-2013[43].

7.3.2 **Освітлення**

Освітлення робочого місця при налагодженні програмного забезпечення здійснюється суміщеним освітленням: природне освітлювання бокове, штучне освітлення загальне рівномірне та комбіноване.

Проведені операції зі створення програмного продукту відносяться до зорової роботи дуже високої точності – розряду II, підрозряд “в” згідно ДБН В.2.5-28-2006[44]

Нормативне значення коефіцієнта природного освітленості (КПО) для II розряду зорових робіт при боковому освітленні складає =1,5%.

Нормативні параметри освітлення та значення КПО при суміщеному освітленні наведено в таблиці 7.3.

Таблиця 7.3 – Нормовані значення КПО при суміщеному освітленні та освітленості на робочих поверхнях при штучному освітленні для приміщень обчислювального центру

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Характе-  ристика зорової роботи | Мінімаль-  ний розмір об’єкту розрізне-  ння, мм | Розряд та під розряд зорової роботи | Контраст з фоном | Фон | Суміще-не освітле-  ння, КПО, ен,  **%** | Штучне освітлення,лк | |
| Комбі-  новане | Загаль-не |
| Дуже високої точності | 0,15–0,3 | IIв | Середній | Середній | 1,5 | 2000 | 500 |

Коефіцієнт природного освітлення КПО обчислюється за формулою (7.1):

, (7.1)

де - нормований коефіцієнт освітленості, для II розряду зорової роботи дорівнює 1,5%;

 - коефіцієнт світлового клімату, рівний 0,9 (тому що вікна виходять на північ);

- номер групи забезпеченості природним світлом (для м. Харкова дорівнює 2).

На підставі формули (7.1) маємо:

Згідно з ДБН В.2.5-28-2006 [44] отримані за формулою значення слід округлити до десятих долей, тому =1,4%.

Реалізація вирахування КПО здійснена при будівництві будинку шляхом створення необхідної площі віконних прорізів.

Відповідно до ДБН В.2.5-28-2006 [44] для штучного загального рівномірного освітлення в приміщенні обчислювального центру підприємства мінімально допустима освітленість =500лк досягнута розрахунком кількості і розміщення ламп ЛБ білого кольору визначеної потужності.

Реалізація штучного освітлення здійснено розрахунковим методом коефіцієнта використання світлового потоку. Необхідний світловий потік однієї лампи визначений за формулою 7.2:

, (7.2)

де - мінімальна освітленість, лк; S – площа приміщення, м;

- коєфіцієнт запасу;

Z – коефіцієнт нерівномірності;

Ф – густина світлового потоку, тм;

n – кількість ламп у світильнику;

N – кількість світильників;

η – коефіцієнт використання світлового потоку, в долях одиниць.

Як світильник обраний ЛПО 4х18 із дзеркальними гратами.

Розміри обчислювального центру 3,5х3х3м. Освітлювана площа S=10,5 .

Обрано систему загального рівномірного освітлення і нормована освітленість =500лк.

Коефіцієнт запасу К прийнятий для приміщень з малими виділеннями пилу і дорівнює 1,5.

Коефіцієнт Z, що враховує нерівномірність освітлення, при використанні люмінесцентних ламп прийнятим 1,1. Для даного типу густина світлового потоку Ф=4960лм. Коефіцієнт відображення поверхонь у приміщенні обчислювального центру дорівнюють: =70%, =50%,=30%.

Знаходимо індекс приміщення за формулою 7.3:

, (7.3)

де A, B, h – довжина,ширина та розрахункова висота (підвіски світильника над робочою поверхнею в метрах) приміщення:

, (7.4)

де Н – геометрична висота приміщення, дорівнює 3м,

- звис світильника (довжина штанги або шнура), дорівнює 0м;

- висота робочої поверхні над рівнем підлоги, дорівнює 0,8м.

Таким чином, дорівнює:

h = 3-0-0,8=2,2.

Індекс приміщення:

.

З таблиці, приведеної в ДБН В.2.5-28-2006 [44], знайдений коефіцієнт використання світлового потоку η для світильників группи 3. Він дорівнює 0,41.

З формули (7.2) визначена необхідна кількість світильників N, за умові, що в кожному світильнику дві лампи (n=2):

.

На рисунку 7.1 приведена схема розташування світильників на плані приміщення.

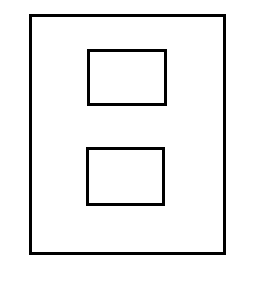


Рисунок 7.1 – План розташування світильників

7.3.3 **Шум і вібрація**

Одним з найбільш поширених факторів зовнішнього середовища, які несприятливо впливають на людину, є шум. На робочих місцях в приміщенні шум створюється установками кондиціонування повітря, перетворювачами напруги, принтерами, високошвидкісними приводами CD-ROM та іншим обладнанням.

Згідно з ГОСТ 12.1.003-83\* [45] в приміщеннях на робочому місці працівника при вирішенні завдань, що вимагають концентрації уваги, рівень шуму не повинен перевищувати 50 дБА. Для зниження рівня шуму використовуються демпфіруючі матеріали.

Рівень вібрації не повинен перевищувати 75 дБ при віброшвидкості згідно ДСТУ ГОСТ 12.1.012:2008 [46], а віброприскорення - 33 дБ.

Основними методами захисту від шуму і вібрації є: зниження шуму й вібрації в джерелі, зниження шуму і вібрації на шляху розповсюдження, застосування індивідуальних засобів захисту, організаційно-профілактичні методи захисту.

7.3.4 **Електромагнітні випромінювання**

Повітря зовнішнього середовища повинне містити позитивні та негативні іони (таблиця 7.4).

Таблиця 7.4 — Рівні іонізації повітря приміщень на робочому місці оператора в приміщенні з ЕОМ

|  |  |  |
| --- | --- | --- |
| Рівні | Кількість іонів в 1см3 повітря | |
|  | Позитивні | Негативні |
| Мінімально необхідні | 400 | 600 |
| Оптимальні | 1500–3000 | 3000–5000 |
| Максимально припустимі | 50 000 | 50 000 |

Для захисту від електромагнітного випромінювання застосовується спеціальне покриття екрана дисплея, використовуються захисні екрани.

Напруга електромагнітних полів у діапазоні 1÷12 кГц, 60÷300 кГц по магнітній і електричній складовій повинні відповідати вимогам до ДСанПіН.3.3.2–007–98 [47] (таблиця 7.5).

Таблиця 7.5 - Допустимі параметри електричних неіонізуючих випромінювань і електростатичного поля

|  |  |  |  |
| --- | --- | --- | --- |
| Види поля | Допустимі параметри поля | | Допустима  поверхнева  щільність потоку  (інтенсивність  потоку енергії),  Вт/м2 |
| За  електричною  складовою  (Е), В/м | За  магнітною  складовою  (Н), А/м |
| 1 | 2 | 3 | 4 |
| Напруженість електромагнітного поля |  |  |  |
| 60 кГц до 3 мГц | 50 | 5 |  |
| 3 кГц до 30 мГц | 20 | – |  |

Закінчення таблиці 7.5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | |
| 30 кГц до 50 мГц | 10 | 0,3 |  | |
| 30 кГц до 300 мГц | 5 | **–** |  | |
| 300 кГц до 300 гГц | **–** | **–** | 10 |
| Електромагнітне поле оптичного діапазону в ультрафіолетовій частині спектру |  |  |  | |
| УФ–С (220–280 мм) | **–** | **–** | 0,001 | |
| УФ–В (280–320 мм) | **–** | **–** | 0,01 | |
| УФ–А (320–400 мм) | **–** | **–** | 10 | |
| в видимій частині спектру:  400–760 мм, |  |  | 10 | |
| в інфрачервоній частині спектру:  0,76–10,0 мм |  |  | 35,0–70,0 | |
| Напруженість електричного поля ВДТ |  |  | 20 кв/м | |

Загальні способи захисту віх впливу електромагнітного випромінювання — це зменшення безпосередньо біля джерела (досягається збільшенням відстані між джерелом та робочим місцем, зменшенням потужності випромінювання).

7.4 **Ергономічні вимоги до робочого місця**

Обладнання та організація робочого місця працюючих з ЕОМ повинні забезпечувати відповідність конструкції всіх елементів робочого місця та їх взаємного розташування ергономічним вимогам згідно ДСанПіН 3.3.2 – 007 – 98 [47] і НПАОП 0.00-1.28-10 [41]. Конструкція робочого місця користувача ЕОМ повинна забезпечити підтримання оптимальної робочої пози. Природне світло повинне падати збоку, зліва. Висота робочої поверхні робочого столу - 720 мм. Ширина 1100 мм, довжина 1000 мм. Такі параметри забезпечують можливість виконання операцій в зоні досяжності моторного поля (згідно ДСанПіН 3.3.2 - 007 - 98 висота - 680 - 800 мм, ширина - 600 - 1400 мм, глибина - 800 - 1000 мм).

Простір для ніг робочого столу має висоту 650 мм , ширину 600 мм , глибину ( на рівні колін ) 470 мм , на рівні витягнутої ноги - 680 мм , що відповідає санітарним правилам і нормам , згідно з якими висота - не менше 600 мм, ширина - не менше 500 мм, глибина - не менше 450 мм, на рівні витягнутої ноги - не менше 650 мм.

Робочий стілець повинен бути підйомно - поворотним , регульованим за висотою. Зусилля регулювання повинна перевищувати 20 Н.

Висота поверхні сидіння в межах 400 - 500 мм. А ширина і глибина становити не менше 400 мм . Кут нахилу сидіння - до 15 град вперед і до 5 град назад.

Висота спинки стільця 350 мм, що не відповідає встановленим нормам 300 ± 20 мм, ширина 400 мм (не менше 380 мм), радіус кривизни горизонтальної площини - 400 мм. Кут нахилу спинки повинен регулюватися в межах 1 - 30 град від вертикального положення. Відстань спинки від переднього краю сидіння в межах 260-400 мм.

Екран ПЕОМ повинен розташовуватися на оптимальній відстані від очей користувача, становить 600-700 мм. Розміщення екрану має забезпечувати зручність зорового спостереження у вертикальній площині під кутом +30 град до нормального лінії погляду працюючого.

Клавіатуру слід розташовувати на поверхні столу на відстані 100-300 мм від краю, зверненого до працюючого. Поверхня клавіатури має бути матовою з коефіцієнтом відображення 0,4. Розміщення пристрою введення - виведення інформації має забезпечувати добру видимість екрану ПЕОМ, зручність ручного керування в зоні досяжності моторного поля і за висотою 900-1300 мм, по ширині 400-500 мм.

Для подолання недоліків слід використовувати пересувну клавіатуру; повинні бути передбачені спеціальні пристосування для регулювання висоти столу, клавіатури і екрану, а також підставка для рук.

У цілому в даному приміщенні ергономічні характеристики відповідають нормам робочого місця, відповідно до ДСанПіН 3.3.2-007-98 [47].

7.5 **Електробезпека**

Приміщення в якому виконувалася робота, відноситься до категорії приміщень з підвищеною небезпечністю ураження людини електричним струмом тому, що існує можливість одночасного дотику людини до металоконструкції будинків, які зв’язані з землею, технологічних апаратів, механізмів чи іншого устаткування, а також до металевих корпусів електроустаткування згідно ПУЭ-87 [48]. По способу захисту людини від ураження електричним струмом ЕОМ, та устаткування для обслуговування, ремонту та наладки ЕОМ повинні відповідати I класу захисту, згідно до ПУЭ-87 [48] та НПАОП 0.00–1.28–10 [41].

При проектуванні систем електропостачання, монтажі силового електроустаткування й електричного висвітлення в будинках і приміщеннях для ЕОМ необхідно дотримуватися вимог нормативно–технічної документації. Комплекс необхідних заходів щодо техніки безпеки визначається, виходячи з видів електроустановки, її номінальної напруги, умов середовища, типу приміщення й доступності електроустаткування.

Персональна ЕОМ є однофазним споживачем електроенергії від трифазної, чотирьох провідної мережі змінного струму з глухозаземленою нейтраллю напругою 380/220В, частотою 50Г.

Документом ПУЭ–87[48] передбачені наступні міри електробезпеки: конструктивні, схемно–конструктивні і експлуатаційні.

Конструктивні заходи: персональна ЕОМ відноситься до електроустановок до 1000 В закритого виконання, всі рубильники встановлені в закритих корпусах, всі струмоведучі частини розміщені в захисних коробах або покриті шаром ізоляції, який виключає можливість дотику до них. Комп'ютер має робочу ізоляцію і елементи заземлення.

Схемно-конструктивні міри: в електричних мережах із глухозаземленою нейтраллю як схемно-конструктивну міру безпеки застосування занулення – навмисне з’єднання металевих не струмоведучих частин комп’ютера, що у випадку аварії можуть виявитися під напругою, з нейтраллю з ГОСТ 12.1.030-81\*[49].

Експлуатаційні заходи:

– монтаж, обслуговування, ремонт і налагодження ЕОМ, заміна деталей, пристроїв, блоків повинна здійснюватися тільки при повному відключені живлення;

– у приміщеннях, де експлуатується більше п’яти комп’ютерів на видному й доступному місці встановлюється аварійний і резервний вимикач для повного відключення електроживлення;

– заземлені конструкції в приміщенні повинні бути надійно захищені діелектричними щитками або сітками від випадкового дотику.

7.6 **Пожежна безпека**

Приміщення з ЕОМ належить до категорії В за пожежонебезпечністю згідно з НАПБ Б 03.002–2007 [50] через присутність у примiщеннi твердих матеріалів, таких як: робочі столи, ізоляція, папір. Відповідно до вимог ДБН В.1.1-7-2002 [51] трьохповерхова будівля, в який виконувалася дана робота, має ступінь вогнестiйкостi II, тому пожежна безпека згідно ГОСТ 12.1.004–91\* [52] забезпечується наступними мірами:

– системою запобігання пожеж (СЗП);

– система протипожежного захисту (СПЗ);

– організаційними заходами щодо пожежної безпеки.

Для зменшення небезпеки утворення в пальному середовищі джерел запалювання передбачено:

1. використання устаткування, що відповідає класу пожежонебезпечної зони П–ІІа відповідно до ПУЭ–87 [48]: ступінь захисту електроапаратури повинна бути не менш ІР–44, ступінь захисту світильників ІР–23, відповідно до ПУЭ–87 [48];
2. блискавковідвід будинків, споруджень і устаткування; для даного класу пожежонебезпеки, зони П–ІІа і місцевості із середньою грозовою діяльністю 20 і більше грозових годин у рік, тобто для умов м. Харкова встановлена III категорія блискавко захисту відповідно до ДСТУ Б В.2.5–38:2008 [53];
3. забезпечення захисту від короткого замикання (контроль ізоляції, використання запобіжників);
4. застосування заземлення захисного екрана для стоку статичної електрики;
5. використання первинних засобів пожежогасіння — вогнегасників ВВК – 5, ВП-5.

Організаційними заходами протипожежної профілактики є:

– навчання виробничого персоналу протипожежним правилам;

– видання необхідних інструкцій, плакатів, засобів наочної агітації, плану евакуації персоналу у випадку пожежі.

8 ЦИВІЛЬНИЙ ЗАХИСТ

Цивільний захист (ЦЗ) – це функція держави, спрямована на захист населення, територій, навколишнього природного середовища та майна від надзвичайних ситуацій шляхом запобігання таким ситуаціям, ліквідації їх наслідків і надання допомоги постраждалим у мирний час та в особливий період [54].

У даному розділі дипломної роботи розглянуто проведення рятівних та інших невідкладних робіт у надзвичайних ситуаціях.

Надзвичайна ситуація — це порушення нормальних умов життя і діяльності людей на об'єкті або території, спричинене аварією, катастрофою, стихійним лихом, епідемією, епізоотією, епіфітотією, великою пожежею, застосовуванням засобів ураження, що призвели або можуть призвести до людських і матеріальних витрат.

Ліквідація наслідків надзвичайних ситуацій — це проведення комплексу заходів, які включають аварійно-рятувальні та інші невідкладні роботи, що здійснюються при виникненні надзвичайних ситуацій техногенного, природного та військового характеру і спрямовані на припинення дії небезпечних факторів, рятування життя та збереження здоров'я людей, а також локалізацію надзвичайних ситуацій [55].

Аварійно-рятувальні роботи — це роботи, спрямовані на пошук, рятування і захист людей, в тому числі надання їм невідкладної медичної допомоги, захист матеріальних і культурних цінностей та довкілля при ліквідації наслідків надзвичайних ситуацій.

Рятувальні роботи з метою врятування людей і надання їм допомоги включають: розвідку району лиха і осередку ураження, маршруту висування формувань та проведення робіт; локалізацію і ліквідацію пожеж на шляху введення рятувальних формувань і об'єктах рятувальних робіт (розшуку і рятування людей, які знаходяться в завалених сховищах, підвалах, завалах, палаючих, загазованих, задимлених або затоплених будинках і виробничих приміщеннях), розкриття розвалених, пошкоджених, завалених захисних споруд і рятування людей, які знаходяться в них; надання першої медичної допомоги потерпілим; винесення потерпілих і евакуація з осередку ураження, небезпечних зон у безпечний район; санітарну обробку людей, ветеринарну обробку сільськогосподарських тварин, знезаражування території, будівель, споруд, продовольства, води, техніки, сировини.

Одночасно або перед рятувальними роботами необхідно виконати інші невідкладні аварійні роботи. Наприклад, для того щоб підвезти людей і техніку, необхідно розчистити завалені проїзди, навести переправи, подати воду для гасіння пожеж тощо.

За організацію і проведення рятувальних та інших невідкладних робіт відповідає керівник ЦЗ об'єкта чи населеного пункту. Він особисто керує підпорядкованими формуваннями через служби ЦЗ.

Безпосередньо на місці проведення рятувальних робіт особовим складом керує командир формування. Він стежить за ходом роботи, за встановленим режимом роботи, за зміною обстановки, проведенням перегрупування чи перестановки сили і засобів на місці роботи, контролює суворе дотримання заходів захисту і безпеки особового складу.

До невідкладних робіт належать: прокладання колонних шляхів і влаштування проїзду, проходів у завалах і зонах забруднення радіоактивні речовини, зараження сильнодіючими ядучими речовинами, локалізація і ліквідація аварій на газових, енергетичних, водопровідних, каналізаційних і технологічних мережах з метою створення умов для проведення рятувальних робіт: укріплення або обвалення пошкоджених і з загрозою обвалу конструкцій будівель і споруд на шляхах руху формувань і в місцях роботи: ремонт і відновлення пошкоджених та зруйнованих ліній зв'язку і комунально-енергетичних мереж з метою забезпечення рятувальних робіт, потреб населення й особового складу формувань, які працюють у районах стихійного лиха, аварії чи осередку ураження, а також для протипожежних заходів [55].

Для ведення рятувальних та інших невідкладних робіт рішенням керівника ЦЗ створюються угруповання ЦЗ. Склад угруповання визначається з врахуванням сил і засобів, характеру й обсягу робіт, які треба виконати.

Рятувальні й невідкладні роботи неможливо провести в короткі строки без використання техніки. Наявну техніку залежно від виду робіт можна розділити на групи: екскаватори, трактори, бульдозери, крани, самоскиди, домкрати, лебідки — для розчищення завалів, піднімання і переміщення вантажів, конструкції будівель і споруд; пневматичні машини — відбійні молотки, бурильні інструменти для подрібнення завалених конструкцій будівель, пробивання отворів, з метою надання повітря або виведення потерпілих; бензорізи, електро- і газозварювальні апарати для розрізання металевих конструкцій; авторемонтні майстерні, станції обслуговування, заправщики паливом, агрегати для освітлення — для ремонту і обслуговування техніки, залученої для проведення рятувальних робіт; насоси, мотопилки, пожежні машини, поливальні машини — для гасіння пожеж і відкачування води; автомобілі вантажні, автобуси, інші транспортні засоби, кінний транспорт — для евакуації потерпілих і тварин із небезпечної зони.

Успішне проведення рятувальних робіт досягається своєчасною організацією і безперервним веденням розвідки, добуванням достовірних даних на встановлений час; високою технічною, морально-психологічною підготовкою, умінням ведення робіт, знанням і суворим дотриманням правил безпеки.

Насамперед потрібно організувати розвідку району лиха надзвичайної ситуації ланками розвідувальної групи, щоб у коротші строки з'ясувати характер і межі руйнування та пожеж і їх ступінь. Висуваючи сили і засоби для проведення робіт, необхідно перш за все влаштувати проїзди і проходи до об'єктів проведення робіт. Для цього застосовують бульдозери, автокрани, грейдери. Ширина проїздів має бути 3,5—4,0 м для одностороннього і 7,0—8,0 м для двостороннього руху, через 150—200 м мають бути роз'їзди довжиною 10,0—20,0 м.

В організації і веденні рятувальних робіт особливе значення мають пошуки потерпілих. Необхідно встановити, де і в яких умовах вони перебувають. Потрібно ретельно обстежити завали, підвальні приміщення, порожнини завалів будівель.

Звільнення людей з-під завалів є найважливішим і найскладнішим видом рятувальних робіт. Якщо потерпілі знаходяться поблизу поверхні або завалені невеликими уламками одноповерхових будівель, то розбирають завали вручну. Потерпілих, які знаходяться в глибині завалів (під завалом), дістають через вузькі проходи (висотою 0,7—0,9 м, шириною 0,6—0,7 м), зроблені з боку завалів. Для прокладання проходів використовують пустоти і щілини, що виникли в завалі від падіння великих елементів будівель. Виносити уражених через зроблений прохід можна на руках, у плащах, брезенті, ковдрі, ношах, волоком.

При руйнуванні великих будівель для розбирання завалів необхідні потужні піднімальні крани, великі екскаватори, пересувні електростанції і ліхтарі для роботи вночі.

Землетруси останніх років показують, що люди під руїнами можуть залишатися живими, якщо вони не поранені, до двох-трьох тижнів.

Для рятування людей із пошкоджених дво-, три- (і більше) поверхових будинків зі зруйновними виходами і сходами споруджують трапи, настил із дощок товщиною не менше 5 см з прибитими впоперек дощок дерев'яними брусками на відстані 25—30 см один від одного, а також роблять отвори в сусідні (суміжні) приміщення, які мають виходи. У ряді випадків для рятування потерпілих з верхніх поверхів напівзруйнованих будинків, коли немає безпосередньої загрози обвалу, застосовують переносні приставні драбини, канати, механічні драбини, підвісні колиски, вишкові машини.

Нестійкі конструкції, падіння яких може викликати небажані наслідки, обвалюють. Конструкції, намічені до обвалювання, тимчасово укріпляють підкосами, розпірками, стояками і огороджують [55].

Перед відкопуванням завалених сховищ і укриттів треба спробувати встановити зв'язок з потерпілими, з'ясувати їх стан. Для цього використовують телефон і радіо, а якщо це неможливо, то перемовляються з людьми, що знаходяться в укриттях, через повітрезабірні отвори, відкриті двері, віконниці, люки, а також вдаються до перестукування по стояках водопостачання чи опалення, які ведуть у підвал. Після встановлення зв'язку з'ясовують забезпеченість людей, що знаходяться в укритті (підвалі), повітрям.

Якщо необхідно подати у сховище, укриття повітря, слід відшукати повітрезабірні отвори, що збереглися, розчистити і через них подавати повітря. При зруйнуванні повітрезабірних каналів, треба відкрити двері чи віконниці аварійного виходу, а якщо і це зробити неможливо, в перекритті чи стінах прорубати отвори, через які за допомогою вентилятора або компресора подати повітря, а також воду, їжу і медикаменти. При загрозі сховищу затоплення чи проникнення газу треба негайно відключити пошкодженні ділянки мереж водопроводу, теплофікації чи газопроводу. Із затоплених приміщень відкачують воду.

Способи відкопування завалених сховищ є різні: розчищають завали над входом або аварійним виходом, пробивають отвори у стінах чи перекриттях, прокладають підземну галерею до стін сховища або пробивають у цій стіні отвори. У підвалах і погребах можна розчищати тільки входи. Потерпілим необхідно надати першу медичну допомогу. Допомога надається медичними формуваннями. Після надання першої медичної допомоги потерпілих направляють на медичні пункти чи в лікарні для надання їм лікарської допомоги.

Із небезпечної території людей необхідно терміново евакуювати в безпечні райони — пішки, а також з допомогою всього наявного транспорту.

Забруднених радіоактивними речовинами або заражених хімічними і біологічними засобами необхідно направити на санітарну обробку.

При руйнуванні водопровідних мереж найбільше пошкоджуються стояки, це може призвести до затоплення сховищ, підвалів або місць, де проводять рятувальні роботи. Слід негайно відключити зруйновані ділянки труб, забивши отвори в трубах дерев'яними пробками, або перекрити засувки (забірні гвинти). У першу чергу відключають засувку з боку насосної станції, яка живить водою, а потім засувку, розташовану з іншого боку пошкодженої ділянки. Перекривати засувку треба повільно, бо гідравлічний удар, що виникає при різкій зупинці руху води, може зруйнувати інші ділянки водопроводу.

Перш ніж спуститись у колодязь необхідно перевірити загазованість бензиновою лампою. Якщо в колодязі є метан або сірководень, полум'я в лампі зменшується, від присутності вуглекислоти, потухне полум'я збільшиться в разі наявності парів ефіру або бензину.

Звільнити колодязі й камери від загазованості можна природним провітрюванням, з допомогою вентилятора або заповненням водою. Якщо неможливо повністю звільнити колодязь від загазованості, роботи можна продовжувати тільки в ізолюючих протигазах.

При гасінні пожеж або в інших випадках аварії на водопровідних мережах необхідно спорудити тимчасові обвідні лінії, поставивши на найближчі пожежні гідранти, стендери з приєднаними до них прядивними рукавами [55].

Таким чином, розглянуте питання дії населення при надзвичайних ситуаціях, важливе і актуальне для збереження життя робітників, швидкого реагування в складних умовах паніки, швидкого проведення рятувальних робіт.

ВИСНОВКИ

У ході роботи було розроблено програмне забезпечення для вирішення задач багатокритеріальної оптимізації генетичними алгоритмами. У ході виконання дипломної роботи була розглянута задача багатокритеріальної оптимізації, поняття Парето-оптимальнсті, та питання використання генетичних алгоритмів стосовно вирішення цих задач. По темі дипломної роботи був зроблений літературний огляд. Досліджений класичний генетичний алгоритм, розглянуті поняття, які він використовує, та основні генетичні оператори. Були розглянуті різноманітні підходи до багатокритеріальної оптимізації генетичними алгоритмами. Вивчені методи, що реалізують ці підходи та їх властивості, а саме заснований на використанні зважених сум метод RWGA, метод незалежної селекції VEGA, заснований на рангах Голдберга метод MOGA, та метод NPGA, що використовує концепцію Парето-домінування та поняття ніші для запобігання передчасної збіжності рішення. Розроблене програмне забезпечення, що реалізує ці методи для вирішення задач багатокритеріальної оптимізації за допомогою генетичних алгоритмів, в тому числі вирішується задача оптимізації токарної обробки титану з субмікро- і нанокристалічною структурою. Була проведена перевірка коректності роботи розробленого додатку за допомогою тестових функцій. Виконаний пошук оптимальних початкових даних для роботи генетичних алгоритмів, а також проведений порівняльній аналіз їх роботи.

Також в роботі було розглянуто економічну частину представлену в огляді існуючих підходів проблеми і представлено метод розробки програмного забезпечення, що дозволяють ефективно зменшити час на виконання задачі, представлено розрахунок витрат на НДР, економічний ефект, економічна ефективність, термін окупності витрат (інвестицій). В результаті проведених розрахунків отримали, що кошторис витрат на науково - дослідницьку роботу становить 39897,00 грн.

З приведеного розділу охорони праці можна зробити висновок, що дотримання всіх вище наведених нормованих параметрів шкідливих виробничих факторів дозволяє забезпечити здорові й безпечні умови роботи. Також в ході виконання роботи було ретельно перевірено ергономічні умови праці робочого місця і виявлено, що вони відповідають необхідним нормам.

За результатами даної роботи можна зробити висновок, що генетичні алгоритми є дуже потужними та перспективними сучасними методами для вирішення широкого спектру задач, у тому числі і задач багатокритеріальної оптимізації.

СПИСОК ДЖЕРЕЛ ІНФОРМАЦІЇ

1. Рутковская Д. Нейронные сети, генетические алгоритмы и нечеткие системы. / Д. Рутковская, М. Пилиньский, Л. Рутковский. – М.: Горячая линия – Телеком, 2006. – 452 с.
2. Гладков Л.А. Генетические алгоритмы. / Л.А. Гладков, В.В. Курейчик, В.М. Курейчик. - М.: ФИЗМАТЛИТ, 2006. — 320 с.
3. Гладков Л.А. Методы генетического поиска. / Л.А. Гладков, Л.А. Зинченко, В.В. Курейчик, В.М. Курейчик, Б.К. Лебедев, Е.В. Нужнов, С.Н. Сорокин. – Таганрог: Изд-во ТРТУ, 2002. – 122 c.
4. Фогель Л. Искусственный интеллект и эволюционное моделирование. / Л. Фогель, А. Уолш. – М.: Мир, 1969. – 228 с.
5. Панченко Т. В. Генетические алгоритмы. / Т. В. Панченко. –Астрахань: Издательский дом «Астраханский университет», 2007. – 87 с.
6. Емельянов В. В. Теория и практика эволюционного моделирования. / В. В. Емельянов, В. М. Курейчик, В. В. Курейчик. - М.: Физматлит, 2003. –432 с.
7. Курейчик В.М. Генетические алгоритмы. / В. М. Курейчик. - Таганрог: изд-во ТРТУ, 1998. - 242 с.
8. Аоки М. Введение в методы оптимизации. / М. Аоки. – М.: Наука. Главная редакция физико-математической литературы, 1977. – 344 с.
9. Беляков Г.П. Основы системотехники: / Г.П. Беляков, В.А. Сарычев, В.А. Сорокин, В.О. Чернышев; под ред. В.О. Чернышева. – Томск: МГП «РАСКО», 1992. ­­– 312 с.
10. Дубов Ю.А. Многокритериальные модели формирования и выбора вариантов систем. / Ю.А. Дубов, С.И. Травкин, В.Н. Якимец. – М.: Наука. Гл. ред. физ.-мат. лит., 1986. – 296 с.
11. Емельянов С.В. Многокритериальные методы принятия решений. / С.В. Емельянов, О.И. Ларичев. – М.: Знание, 1985. – 32 с.
12. Кини Р.Л. Принятие решений при многих критериях: предпочтения и замещения. / Р.Л. Кини, Х. Райфа. – М.: Радио и связь, 1981. – 560 с.
13. Лесин В. В. Основы методов оптимизации. / В. В. Лесин, Ю. П. Лисовец. ­– М.: Изд-во МАИ, 1995. ­ – 344 с.
14. Подиновский В.В. Парето-оптимальные решения многокритериальных задач. / В.В. Подиновский, В.Д. Ногин. – М.: Наука. Главная редакция физико-математической литературы, 1982. – 256 с.
15. Подиновский В.В. Оптимизация по последовательно применяемым критериям. / В.В. Подиновский, В.М. Гаврилов. - М.: «Сов. радио», 1975. -­ 192 с.
16. Семенкин Е.С. Оптимизация технических систем. / Е.С. Семенкин, О.Э. Семенкина, С.П. Коробейников. – Красноярск: СИБУП, 1996.­ – 284 с.
17. Прохоренок Н. HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера. / Н. Прохоренок. - Москва: БХВ-Петербург, 2010. – 912 с.
18. Коэн Л.И. Полный справочник по HTML, CSS и JavaScript. / Л.И. Коэн, Д.И. Коэн. - Москва: ЭКОМ Паблишерз, 2007. – 1168 с.
19. Дронов В. JavaScript и AJAX в Web-дизайне. / В. Дронов. - Санкт-Петербург: БХВ-Петербург, 2008. – 736 с.
20. Дунаев В. Сценарии для Web-сайта. PHP и JavaScript. / В. Дунаев. - Санкт-Петербург: БХВ-Петербург, 2008. – 576 с.
21. Никсон Р. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript. / Р. Никсон. - Санкт-Петербург: Питер, 2011. – 496 с.
22. Богатырёв М.Ю. Генетические алгоритмы: принципы работы, моделирование, применение. / М.Ю. Богатырёв. – Тула: ТулГУ, 2003. – 152 с.
23. Батищев, Д.И. Генетические алгоритмы решения экстремальных задач. / Д.И. Батищев. - Нижний Новгород: Нижегородский госуниверситет, 1995. - 62с.
24. Тим Джонс, М. Программирование искусственного интеллекта в приложениях. / М. Тим Джонс. – М.: ДМК Пресс, 2004. – 312 с.
25. Вороновский, Г.К. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности. / Г.К. Вороновский, К.В. Махотило, С.Н. Петрашев, С.А. Сергеев. – Харьков: ОСНОВА, 1997. – 112с.
26. Редько В. Г. Эволюционная кибернетика. / В. Г. Редько. - М.: Наука, 2001. – 159 с.
27. Букатова И.Л. Эвоинформатика. Теория и практика эволюционного моделирования. / И. Л. Букатова, Ю.И. Михасев, А.М. Шаров. - М.: Наука, 1991. - 206 с.
28. Букатова, И.Л. Эволюционное моделирование и его приложения. / И. Л. Букатова. - М.: Наука, 1979. - 232 с.
29. Аверченков, В.И. Эволюционное моделирование и его применение: монография. / В.И. Аверченков, П.В. Казаков. - Брянск: БГТУ, 2009. – 200 с.
30. Coello C.A. An empirical study of evolutionary techniques for multiobjective optimization in engineering design. / C.A. Coello. - New Orleans: Department of computer science, Tulane university, 1996.
31. Coello C.A. A comprehensive survey of evolutionary-based multiobjective optimization techniques. / C.A. Coello. - New Orleans: Department of computer science, Tulane university, 1997.
32. Fonseca C.M. Multiobjective optimization and multiple constraint handling with evolutionary algorithms. Part I: A unified formulation. / C.M. Fonseca, P.J. Fleming. - Sheffield: University of Sheffield, 1995.
33. Fonseca C.M. Multiobjective optimization and multiple constraint handling with evolutionary algorithms. Part II: Application example. / C.M. Fonseca, P.J. Fleming. - Sheffield: University of Sheffield, 1995.
34. Holland J.H. Adaptation in Natural and Artificial Systems. / J.H. Holland. - Cambridge: "MA: MIT Press", 1992.
35. Horn J. A niched Pareto genetic algorithm for multiobjective optimization. / J. Horn, N. Nafpliotis, D. E. Goldberg. - Piscataway, 1994.
36. Schaffer J.D. Multiple objective optimization with vector evaluated genetic algorithms. / J.D. Schaffer. - Pittsburgh: PA, 1985.
37. Schwefel H. Evolution and optimum searching. / H. Schwefel. - New York: Wiley Interscience, 1995.
38. Michalewicz, Z. Genetic Algorithms + Data Structures = Evolution Programs. / Z. Michalewicz. - Spingler, 1996. - 387 p.
39. Закон України «Про охорону праці» від 21.11.2002р.
40. Кодекс законів про працю України.
41. НПАОП 0.00-1.28-10. Правила охорони праці під час експлуатації електронно-обчислювальних машин. – К., 2010.
42. ГОСТ 12.0.003-74\* ССБТ Опасные и вредные производственные факторы. Классификация. – Введ. 01.01.1976г.
43. ДБН В.2.5-67:2013. Інженерне обладнання будинків і споруд. Опалення, вентиляція та кондиціонування – К.: Мінрегіон України, 2013. –   140 с.
44. ДБН В.2.5-28-2006 Державні будівельні норми. Інженерне обладнання будинків і споруд. Природне і штучне освітлення. – К., 2007. – 80 с.
45. ГОСТ 12.1.003-83\* ССБТ. Шум. Общие требования безопасности. - Введ.01.07.90 г
46. ДСТУ ГОСТ 12.1012:2008. Вібраційна безпека. Загальні вимоги-Введенний 2008 р.
47. ДСанПіН 3.3.2-007-98. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин. -К., 1998.
48. ПУЭ–87. Правила устройства електроустановок. - М.: Энергоатомнздат, 1988. – 648 с.
49. ГОСТ 12.1.030-81\* Электробезопасность. Защитное заземление, зануление. - Введ. 01.01.82 г. - Изменен 1987 г.
50. НАПБ Б. 03.002-2007. Нормативний акт пожежної безпеки. Норми визначення категорії приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою. Наказ МНС від 03.12.07 №633.
51. ДБН В. 1.1-7-2002. Державні будівельні норми. Захист від пожежі. Пожежна безпека об'єктів будівництва.- К.: Мінбуд України, 2002. - 41с.
52. ГОСТ 12.1.004-91\* ССБТ. Пожарная безопасность. Общие требования. - Введ. 01.07.91 г.
53. ДСТУ Б В.2.5-38:2008. Улаштування блискавкозахисту будівель і споруд. - Київ. МІНРЕГІОНБУД України. НАКАЗ Міністерства регіонального розвитку та будівництва України від 27.06.2008 №269, чинний з 2009-01-01.
54. Кодекс цивільного захисту України - ВРУ № 5403-VI, від 02.10.2012 р.
55. Кулаков М.А. Цивільна оборона. / М.А. Кулаков, В.О. Ляпун, В.О. Мягкий. - Харків: Факт, 2008 - 312 с.