УДК 004.021

Р.З. Кривий, М.М. Лобур, С.П. Ткаченко
Національний університет “Львівська політехніка”,
кафедра систем автоматизованого проектування

РОЗРОБКА ПІДСИСТЕМИ ДЛЯ ДОСЛІДЖЕННЯ ГЕНЕТИЧНИХ АЛГОРИТМІВ З ВИКОРИСТАННЯМ ШАБЛОНІВ

Розглянуто особливості програмної реалізації підсистем для дослідження генетичних алгоритмів. Описано роботу, розроблено підсистеми і виконано аналіз її ефективності.
Ключові слова – генетичний алгоритм, шаблон

The features of subsystems program implementation are considered for research of genetic algorithm. The work of developed subsystem is described and analyzed its effectiveness.
Keywords – genetic algorithm, template

Вступ

Генетичні алгоритми є важливою складовою еволюційних методів. У США і Європі спостерігається бум розвитку генетичних алгоритмів. Відбулись десятки міжнародних конференцій, опублікована величезна кількість літератури по генетичних алгоритмах, і інтерес до цієї тематики постійно зростає.

Генетичні алгоритми – це процедури пошуку, засновані на механізмі природного відбору і наслідування. Вони відрізняються від інших оптимізуючих і пошукових процедур такими ознаками [1]:

- обробляють не значення параметрів самої задачі, а їх закодовану форму;
- здійснюють пошук рішень, враховуючи не одиничну точку, а їх деяку популяцію;
- використовують тільки цільову функцію, а не іншу додаткову інформацію;
- використовують ймовірність, а не детерміновані правила вибору;
- у процесі еволюції кожна нова популяція залежить тільки від попередньої.

Генетичні алгоритми в різних формах застосовуються до вирішення багатьох наукових і технічних проблем, а саме: вони використовуються при створенні різних обчислювальних структур, наприклад, автоматів або мереж сортування; при машинному навчанні; при проектуванні нейронних мереж або керуванні роботами.

Проте, можливо найпопулярніше застосування генетичних алгоритмів – оптимізація багатопараметричних функцій. Багато реальних задач можуть бути сформульовані як пошук оптимального значення, де значення – складна функція, що залежить від певних вхідних параметрів. У деяких випадках потрібно знайти ті значення параметрів, при яких досягається найкраще значення функції. В інших випадках глобальний екстремум не потрібний – рішенням може вважатися будь-яке значення, краще за певну задану величину. У цьому випадку генетичні алгоритми – часто найкращий метод для пошуку "прийнятних" значень. Сила генетичного алгоритму полягає в його здатності маніпулювати одночасно багатьма параметрами, що використовується в сотнях прикладних програм, а також проектування літаків, налаштування параметрів алгоритмів і для пошуку стійких станів систем нелінійних диференціальних рівнянь.

Шаблони в генетичних алгоритмах

Поняття “шаблон” було введено для визначення множини хромосом, які мають деякі загальні властивості. Шаблоном називають рядок вигляду

$$(a1, a2, ..., ai, ..., al),\ ai \in \{0, 1, *\}.$$

Символом "*" в деякому розряді позначається те, що там може бути як 1, так і 0. Наприклад, для двох бінарних рядків "111000111000" і "110011001100" шаблон виглядатиме так: "11*0****1*00".

Тобто, за допомогою шаблонів можна як би виділяти загальні ділянки двійкових рядків і маскувати відмінності. Маючи у складі шаблону m символів "*", можна закодувати (узагальнити) 2m двійкових рядків. Так, наприклад, шаблон "01*0*1" описує набір рядків [2].

{"010001", "010011", "011001", "011011"}.

Особині, якій належить хромосома, що містить набір генів-параметрів завдання, поставлена у відповідність величина, пристосованість, що характеризує її. Оскільки шаблон є узагальненням декількох бінарних хромосом, то особини, що мають хромосоми, які відповідають одному шаблону, пристосованіші. Тобто, за допомогою шаблонів будуть відбиратися кращі рішення, отже, більші значення цільової функції. Наприклад, шаблон для функції f(x)=10+x*sin(x) має такий вигляд (рис. 1).

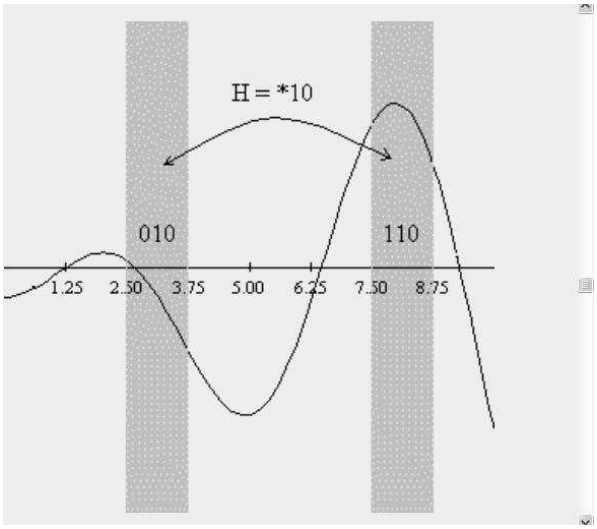

*Рис. 1. Шаблон для функції f(x)=10+x*sin(x)*

Існуючі програмні реалізації генетичного алгоритму

Незважаючи на величезний інтерес до галузі еволюційного моделювання, кількість реально працюючих програм в цій галузі доволі мала. Роботи в цій галузі можна розділити на декілька великих категорій.

Перша категорія програм – пакети, що реалізують класичний генетичний алгоритм з можливим настроюванням параметрів управління основними операторами ГА. Модель хромосоми в таких пакетах має, як правило, стандартну бінарну структуру, а функція відбору задана одним математичним виразом. До таких програм належать: пакет Evolver 4.0 компанії Palisade Corp, пакет GeneHunter компанії Ward System Group тощо. Істотним недоліком цих методів є прив'язка до бінарної або числової моделі хромосоми (хромосома не може мати складної структури). До другої категорії програм по генетичних алгоритмах належать спеціалізовані програми, призначені для вирішення конкретних завдань. Ці генетичні алгоритми розроблені і оптимізовані для вирішення вузької, чітко визначеної проблеми. Третя категорія розробок по генетичних алгоритмах включає суто наукові дослідження, що полягають в дослідженні властивостей і характеристик різних генетичних алгоритмів, їх збіжності і виродженості.

Ефективність генетичного алгоритму при розв’язанні конкретної задачі залежить від багатьох чинників і, зокрема, від таких, як генетичні оператори і вибір відповідних значень параметрів, а також способу представлення рішення на хромосомі. Оптимізація цих чинників приводить до підвищення швидкості і стійкості пошуку, що істотно впливає на застосування генетичних алгоритмів. Тому доцільним є створення і дослідження програм, що використовують шаблони.

Створення початкової популяції

Для представлення хромосоми використано наступний метод. Нехай необхідно представити розв’язки з певного проміжку [A,B] із заданою точністю. У такому разі необхідно розбити наш проміжок на рівні частинки (маленькі проміжки), число яких повинно бути не менше за

(B-A)*(1/eps). Визначивши число проміжків ми можемо знайти число біт, потрібних для кодування такого числа, і після цього ми можемо присвоїти кожному проміжку порядковий номер, який і буде зашифровано всередині хромосоми. У випадку, коли у нас буде багато параметрів, ми можемо збільшити розмір хромосоми у 2,3,… рази відповідно до кількості параметрів. Тоді перша частина відповідатиме за значення першого параметра, друга частина за значення другого і так далі.

Наступне питання – це створення початкової популяції. Нині найвідомішими і найвживанішими на практиці є 3 способи створення початкової популяції.

Стратегія “ковдри” – формування повної популяції, що містить всі можливі рішення. Наприклад, для n-розрядної хромосоми існує 2n варіантів рішень, які складають повну популяцію. Стратегія “дробовика” – генерація достатньо великої випадкової множини рішень. Стратегія “фокусування” – генерація підмножини рішень, що включає різноманіття одного рішення [1].

Перший підхід практично неможливо реалізувати, навіть для задач середньої розмірності, через величезні втрати пам’яті та процесорного часу. Також варто зауважити, що тоді початкова популяція не може розвиватися, оскільки в ній вже міститиметься найкращий розв’язок. Третій спосіб використовується тоді, коли є гіпотеза, що оптимальне рішення є варіацією відомого рішення. У цьому випадку час пошуку оптимального рішення різко зменшується, оскільки алгоритм розпочинає свою роботу в області оптимуму.

Найчастіше застосовують другий спосіб. У цьому випадку в результаті еволюції є можливість перейти в інші області пошуку, і кожна з них має порівняно невеликий простір пошуку. Загалом ефективність генетичного алгоритму, якість рішень і подальша еволюція значною мірою визначаються структурою і якістю початкової популяції. Крім того, застосовують комбінацію другого і третього способів. Спочатку визначаються особи з високим значенням цільової функції, а потім випадково формуються початкові рішення в цих областях.

Оскільки підсистема орієнтована на роботу з певною кількістю параметрів та пошуку оптимального розв’язку з великою точністю, то неможливо реалізувати першу стратегію через брак пам'яті. Третя стратегія теж не підходить, оскільки в такому разі необхідно проводити додаткові дослідження функції, що вже виходить за межі поняття генетичного алгоритму. Через великі розміри хромосоми неможливо здійснити генерацію одразу цілої хромосоми, тому генерація відбувається побітово для кожної хромосоми.

Робота з програмою

Для реалізації підсистеми побудови генетичних алгоритмів з використанням шаблонів вибрано мову програмування C++. Вона дозволяє максимально компактно та ефективно оперувати даними, будувати великі абстрактні моделі та одночасно доступатися до низьких рівнів програмування, що дає змогу повністю контролювати процес реалізації та налагодження алгоритму на всіх його етапах.

Вибір цієї алгоритмічної мови також пов'язаний з найефективнішою підтримкою програмних інтерфейсів, необхідних для реалізації окремих функцій майбутньої системи. Крім того, алгоритмічна мова забезпечена зручним у користуванні середовищем розроблення програм, яке дозволяє легко уникати помилок на етапі розроблення, а також містить велику довідкову систему, яка допомагає у процесі реалізації системи. І, насамкінець, мова дає змогу легко переходити на інші алгоритмічні мови різних рівнів без втрат часу на налаштування великої кількості параметрів, що економить ресурси програміста.

Інтерфейс програми побудований з використанням api MFC (Microsoft Foundation Code). Але ці бібліотеки не передбачають жодних засобів для побудови блоків і графіків, які застосовуються у розробленій програмі.

При запуску розробленої підсистеми перед користувачем з’являється головне вікно програми. Ліворуч розміщене меню, призначене для створення нового блока – елемента генетичного алгоритму. Нижче розміщені форми для вводу вхідних даних. Праворуч розміщене вікно вигляду генетичного алгоритму. У статусному рядку відображається додаткова інформація – програмні коди блоків або допоміжні координати. Створення блока популяції, відбувається виділенням потрібної популяції і перенесенням в робочу область, де пізніше задаються їх параметри. Інші види блоків створюються аналогічно. Після чого всі блоки з’єднуються (рис.2.).

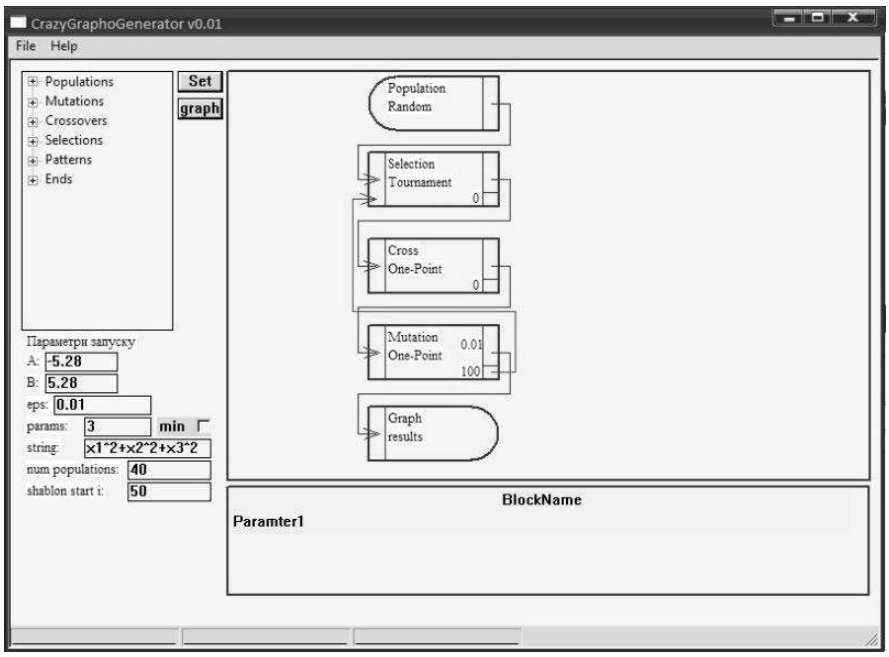

Рис. 2. Головне вікно підсистеми з побудованим генетичним алгоритмом

Коли алгоритм побудовано і параметри задані, здійснюється запуск. При цьому зчитаються дані, проаналізується створений алгоритм і, якщо не буде знайдено помилок, відобразиться результат. При неправильній побудові алгоритму програма попередить користувача про помилки. Після опрацювання генетичного алгоритму результат – масив значень певної довжини – записується у файл для можливості подальшого оброблення і аналізу та відображається на екрані графіком (рис.3). Його ми записуємо у файл. У підсистемі використовується перемикання між відображенням графіка і робочою областю. При невиконаному алгоритмі графік не відображається.

Аналіз результатів та висновки

Для того, щоб перевірити правильність роботи системи, було виконано мінімізацію значення функції з двома параметрами:

$$f(x)=|\sin(x1)|+x2^2+1 \qquad x1,x2 \in [-10,10]$$

Ця функція має декілька глобальних мінімумів – 1, це виконується за умови, коли значення першого параметра близьке до πn/2, а другий параметр дорівнює 0.

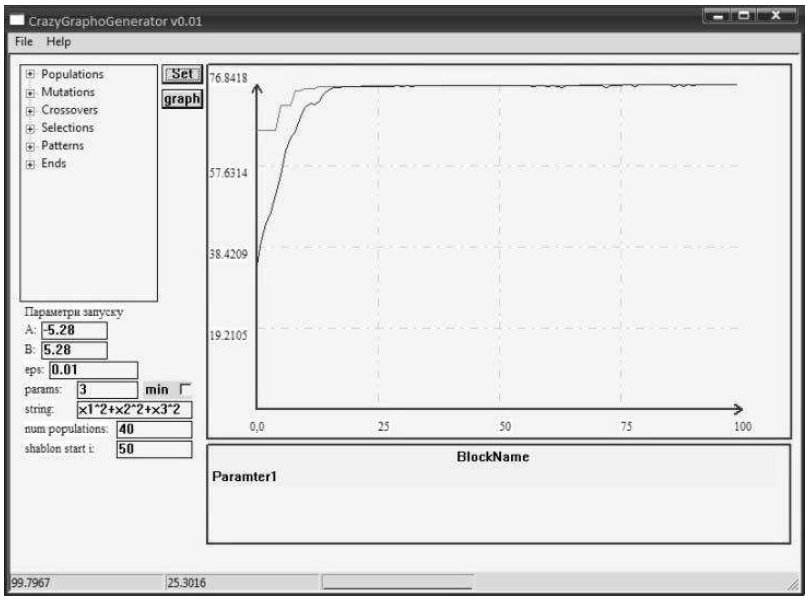

Рис. 3. Графічне відображення результатів

Генетичний алгоритм складемо з таких блоків: створення популяції (метод «дробовика»); селекція (метод «відрізання»); двоточкове схрещування; одноточкова мутація; вивід результатів. Задана ймовірність мутації була доволі низькою 0.5 %. Значення кожного параметра обчислюватимемо з точністю 10^{-5} і використаємо популяцію з 50 хромосом (рис. 4).

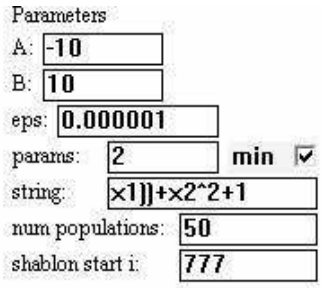

Рис.4. Занесення параметрів генетичного алгоритму

Після запуску алгоритму отримуємо наступний графік (рис. 5).

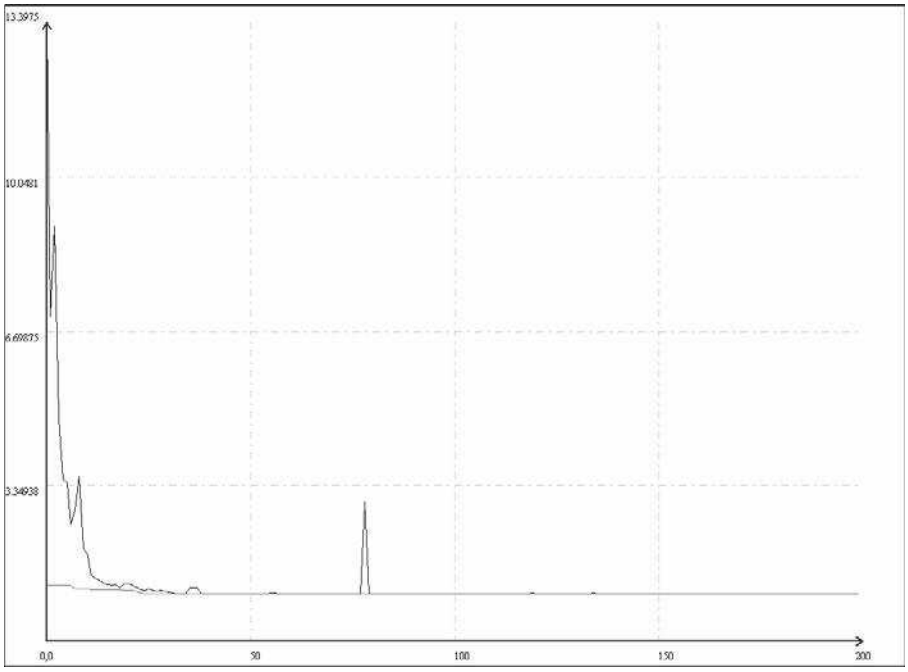

Рис. 5. Результат роботи тестового завдання

Виконавши подальшу серію тестових задач, як з використанням шаблонів, так і без них, було визначено, що правильне використання шаблону призводить до позитивного результату розв'язання поставленої задачі. Підключивши в певний момент використання шаблонів, отримуємо той результат, що операції мутації і схрещування не зможуть погіршувати значення хромосом.

1. Скобцов Ю.О. Основы эволюционных вычислений: Учебн. пособие. – Донецк: ДонНТУ, 2008. – 326 с. 2. Rostyslav Kryvyy, Mykhaylo Lobur, Serhiy Tkatchenko, Yuri Darnobyt: Possibilities of the use of patterns in MEMS design. XVI Ukrainian-Polish Conference on "CAD in Machinery Design. Implementation and Educational Problems", Lviv, 2008, p.45-46