УДК 519.254:004.021

Я.М. Грицишин, Д.В. Корпильов, Р.З. Кривий, Т.В. Свірідова, С.П. Ткаченко
Національний університет "Львівська політехніка"
кафедра систем автоматизованого проектування

ГЕНЕТИЧНІ АЛГОРИТМИ ДЛЯ РОЗВ'ЯЗАННЯ ЗАДАЧ РОЗМІЩЕННЯ

Описано головні аспекти використання генетичних алгоритмів для розроблення гібридних інтегральних схем, функціонування генетичних алгоритмів для автоматизованого розташування об'єктів довільної форми на площині довільної форми. Запропоновано критерії для визначення послідовності вибору лекал та площин для розташування, а також критерії для вибору оптимального розміщення одного лекала. Розроблено генетичний алгоритм для оперування критеріями вибору і відбору необхідної популяції рішень.

In this paper main aspects of using of genetic algorithms in design of hybrid integrated circuits is discussed and described the functioning of genetic algorithm for the automated arranging the arbitrary shape objects on the arbitrary shape platforms. The set of criteria for determination the sequence of selecting templates and platforms for arranging and also a set of criteria for selecting the optimum arranging of single template are suggested. The genetic algorithm for the selecting criteria manipulation and choice of necessary decisions is developed.

Вступ

Групу алгоритмів, що використовують ідею Дарвіна як основну про еволюцію називають генетичними алгоритмами. Вони можуть ділитися на такі напрями: генетичні алгоритми, стратегії еволюції, генетичне програмування, еволюційне програмування.

Генетичні алгоритми використовуються для вирішення таких задач: пошук глобального екстремулу для: багатопараметричних і наближених функцій, задач для знаходження найкоротшого шляху, оптимального розміщення, налаштування нейронних систем, розроблення ігрових стратегій.

Фактично генетичні алгоритми оптимізують значення багатопараметричних функцій, тому їх галузь використання така широка. Всі представлені задачі формуються як функції, які залежать від деякої кількості параметрів і глобальний максимум яких відповідатиме розв'язку задачі [1].

1. Аналіз методів пошуку оптимального плану розкрою

Можна виділити три основні групи методів, які дають змогу знайти розв'язок задачі розкрою:

- методи математичного програмування;
- евристичні методи;
- методи еволюційного пошуку.

До першої групи методів належать методи лінійного і нелінійного програмування. Як правило, вони гарантують знаходження оптимального рішення, але недоліком цих методів є їх висока обчислювальна складність, яка експоненціально зростає із зростанням розмірності завдання розкрою[2, 3, 4]. З цієї причини ці методи не зовсім підходять для вирішення завдання.

Евристичні методи дають змогу значно скоротити часові витрати на пошук розв'язку практично будь-якої задачі розкрою, проте за їх допомогою рідко знаходять оптимальне рішення. Зазвичай за допомогою цих методів знаходиться рішення, близьке до оптимального. Безперечною

перевагою цих методів є можливість врахування додаткового набору технологічних обмежень, правда, це звужує і область використання подібних алгоритмів. До того ж, ці методи не захищені від потрапляння в “пастки локального оптимуму” [4].

Порівняно новими виглядають методи еволюційного пошуку: генетичні алгоритми і методи генетичного програмування (рис. 1). Ці методи успішно поєднують в собі переваги евристичних методів (швидкий пошук розв’язку близького до оптимального), а передбачають способи виходу з локальних екстремумів [5–7].

У класичному генетичному алгоритмі початкова популяція створюється випадковою технікою. Розмір популяції (всю кількість хромосом позначимо як N), який не змінюється протягом всього алгоритму, є фіксований. Кожна особина генерується як L – бітний рядок, де L – довжина коду особини. Кожна особина – це вирішення поточної проблеми. Більш придатна особина – це придатний розв’язок. Ці особливості роблять генетичний алгоритм порівняно з іншими оптимізаційними алгоритмами таким, що дає кращий результат.

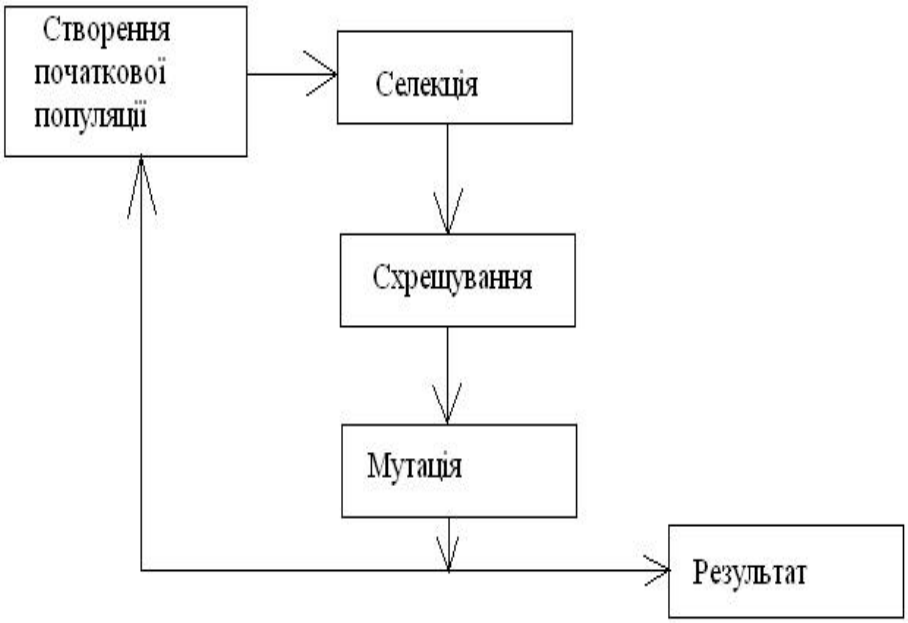

Рис.1. Схема роботи генетичного алгоритму

Алгоритм складається з трьох стадій: створення проміжної популяції вибором від поточного покоління, повторне створення проміжного покоління шляхом кросинговеру, який призводить до формування нової генерації, і мутація.

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

Сила генетичного алгоритму полягає в його здатності маніпулювати одночасно багатьма параметрами, що використовується в сотнях прикладних програм, зокрема проектування літаків, налаштування параметрів алгоритмів і пошуку стійких станів систем нелінійних диференціальних рівнянь.

Як правило, генетичні алгоритми “помиляються” не більше ніж на 5–10%, гарантуючи, для комбінаторних завдань високу швидкість роботи. На практиці генетичні алгоритми нерідко використовують спільно з іншими методами, які дають змогу підвищити їх ефективність [7].

2. Генетичний алгоритм розташування лекала на контурі довільної форми

Вхідними даними для розв'язування задачі розміщення довільних об'єктів на площині є масив лекал, які необхідно розмістити, і масив контурів, в яких потрібно розташувати лекала. Самі контури можуть бути як прямокутної, так і довільної форми, тобто матеріал міг використовуватися в попередніх задачах і залишився у відходах. Так само і лекала – можуть мати правильну і неправильну форму. Насамперед необхідно визначити, які лекала можна буде розташовувати і в якому контурі. У випадку, коли у деякому контурі неможливо розмістити жодне лекало, контур автоматично потрапляє у відходи, і ми його видаляємо з масиву контурів.

Для роботи генетичного алгоритму насамперед нам потрібно визначити, за якими критеріями необхідно вибирати лекала і контури, в які вони будуть розміщуватися. Оскільки наперед невідомо, якої форми в нас будуть лекала і контури площин, на які ми будем розміщувати лекала, необхідно розглядати різні критерії. Спочатку треба розглядати критерії, які будуть опиратися на площу і форму лекал, хоча не треба відкидати і інші можливі критерії.

Визначення таких критеріїв дасть змогу зробити рейтингування критеріїв за ефективністю їх використання з подальшим врахуваннях цієї ефективності у експертній підсистемі.

Для розташування лекал за контурами на основі генетичних алгоритмів було розроблено алгоритм розташування лекала за контуром довільної форми. Було розглянуто такі критерії вибору лекал для розміщення їх на площинах :

- максимальна площа;
- мінімальна площа;
- мінімальний периметр;
- максимальний периметр;
- мінімальне описане коло;
- максимальне описане коло;
- мінімальна описана площа прямокутника;
- максимальна описана площа прямокутника.

Цей список не претендує на повноту, але ці критерії повинні бути базовими для розв'язання задач розкрою за допомогою генетичних алгоритмів.

Наприклад, такий критерій, як максимальна площа лекала дасть змогу відразу розмістити найбільше лекало і зменшить можливість того, що це лекало не буде розміщене. А мінімальна площа лекала дасть можливість розмістити найменші лекала в таких місцях контура, де великі не помістяться, що дасть змогу зменшити відходи.

Залежно від різновидів задач розкрою, розробники можуть додавати інші критерії, які, на їхню думку, можуть істотно вплинути на результат виконання поставленої задачі.

Ці критерії також підходять і для вибору контурів. Для роботи алгоритму необхідно ввести критерії для розміщення лекала в контурі, а саме:

- кількість спільних сторін лекала і площини (цей критерій дає можливість розмістити лекало впритул до контура площини і тим самим зменшити можливі втрати матеріалу);
- кількість утворених контурів після розміщення лекала;
- кількість непридатних контурів, які після розміщення лекала будуть відкинуті (цей критерій дає можливість вибрати таке розміщення, за якого утвориться найменша кількість контурів).

До цих розглянутих критеріїв можуть додатися і інші, які будуть давати позитивний результат для певного різновиду задач розкрою.

Для роботи генетичного алгоритму, створюється початкова популяція. Популяція – це набір генів. У нашому випадку ген – це набір критеріїв для розміщення лекал на площині.

Наприклад: (a,b,c) – це ген,

де a – критерій вибору лекала;

b – критерій вибору контура, на який буде розміщене лекало;

c – критерій вибору розміщення лекал.

Початкова популяція може створюватись як зі всіх можливих генів, так і мати певні обмеження. Тобто, є можливість існування тільки певної кількості генів. Після утворення початкового покоління кожен ген оцінюється на можливість продовження роду, тобто обчислюється функція корисності. У нашому випадку ми оцінюємо, за допомогою якого гена в задачі розкрою буде отримана найменша площа відходів. Оцінивши результати –відбираємо приблизно 30% генів, які показали найкращий результат. Цей процес називається селекцією. Із збережених генів (проміжне покоління) за допомогою генетичних операцій – схрещення і/або мутації – ми утворюємо нове покоління і оцінюємо його спроможність щодо подальшого існування.

Для задач розкрою пошук в досконалих генів продовжується доти, доки не розмістяться всі лекала або не закінчаться контури (рис.2).

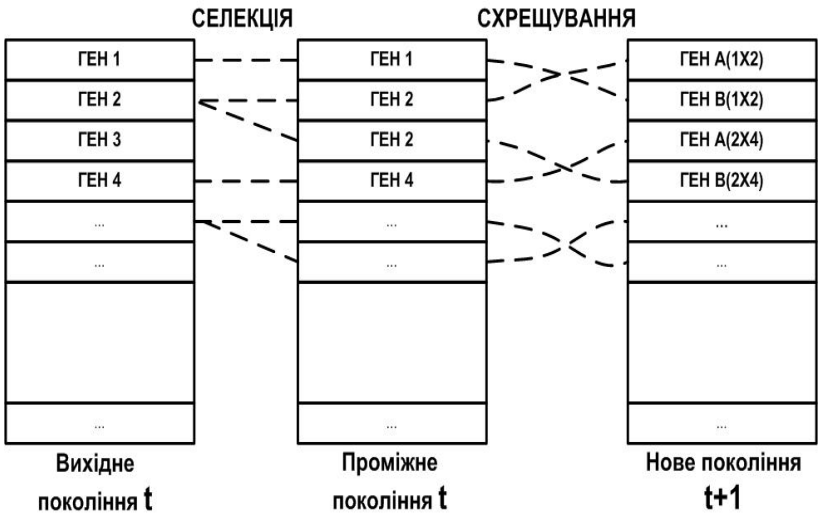

Рис.2. Схема пошуку генів для нової популяції

Вхідними даними для роботи розробленого алгоритму розташування лекал довільної форми на площині є масив лекал, які необхідно розмістити, і масив контурів, в яких потрібно розташувати лекала.

1. Початок роботи алгоритму.
2. Отримання вхідних даних.

Отримується масив лекал, які необхідно розмістити, і масив контурів, в яких розміщуються лекала. Дані передаються з інтерактивної підсистеми.

3. Розраховуємо, які лекала потрапляють до певних контурів.

Аналізуємо, які лекала потрапляють до певних контурів і складаємо список. Якщо в певний контур неможливо розмістити жодного лекала – цей контур автоматично потрапляє у відходи, і ми його видаляємо з масиву контурів.

4. Для кожного генома проводимо ітерацію.

Для всіх можливих варіантів наборів критеріїв (генів) проводимо ітерацію (розташовуємо лекала).

5. Розраховуємо площу відходів і вибираємо m% найкращих результатів.

Для кожного можливого розташування лекала обчислюємо площу контурів, в які вже не потрапить жодне лекало, тобто ті контури, які підуть у відходи. Після цього з отриманих результатів вибираємо m% результатів (селекціонуємо найкращі гени), де площа відходів найменша, враховуючи результати, які наближаються до тих m%.

6. Розраховуємо, які лекала потрапляють до нових контурів.

Аналізуємо, які лекала потрапляють до нових контурів і складаємо новий список лекал.

7. Чи є для утворених нових контурів лекала?

Якщо в нових контурах не можливо розмістити жодного лекала, переходим до кроку 8, в іншому випадку переходимо до кроку 9.

8. Утворені контури відносимо до відходів.

Контури потрапляють до відходів і ми їх видаляємо з масиву контурів.

9. Чи є ще лекала для розміщення?

Перевіряємо, чи є у вихідному масив ще не розміщені лекала. Якщо немає, переходимо до кроку 11. В іншому випадку переходимо до кроку 10.

10. Схрещуємо найкращі результати і отримуємо нові можливі геноми.

За допомогою генетичних операторів (кросинговеру, мутації та інших) схрещуємо отримані покоління для можливого зародження досконалішого гена. Надалідля виконання генетичних операцій не використовуємо тих критеріїв, які дали негативні результати. Отримуємо нові можливі геноми (нову популяцію). Переходимо до кроку 4.

11. Виведення результатів.

На цьому кроці виводимо такі результати:

- послідовность розташування лекал у контурах з координатами розміщення лекал ;
- процент відходів;
- які лекала не розміщені (якщо такі є).

12. Кінець роботи алгоритму.

3. Генетичний алгоритм гібридних інтегральних схем

Задача розміщення вершин графу є зв’язаною з технічною задачею, що проявляється під час проектування гібридних інтегральних схем – розміщення кристалів. Це одна із задач, яка має на меті якісне проектування схем. Тому задача розміщення кристалів передбачає велику кількість модулів для розміщення з мінімальною сумою розміщення прямокутників і мінімальною довжиною трас, які з’єднують модулі. Тобто це задача оптимізації, яку можна розв’язати з використанням генетичних алгоритмів.

Отримання оптимального результату задачі розміщення і стиснення топології можливе завдяки керуванням процесу генетичного пошуку, зміни розміру популяції, розміру покоління, ймовірності використання генетичних операторів, штучної міграції, керування вибором. Цей клас задач належить до задач NP складності. Описана евристика дає змогу знайти значну кількість локальних розв’язків з великою можливістю знаходження глобальних оптимальних результатів [1].

Генетичні алгоритми широко використовуються в суміжній області пошуку оптимального розташування елементів на кристалі. Наприклад, Генетичний Алгоритм для оптимізації області загальної топологічної структури (GALO) заснований на спеціальному кодуванні рішення і визначенні функції оцінки; у ньому застосовуються ефективні оператори схрещування і мутації, а також два додаткові евристичні оператори, призначені для підвищення ефективності методу. Адаптивний підхід автоматично забезпечує оптимальні значення для вірогідності активації операторів ГА. Перший з евристичних операторів вибирає випадковий блок і змінює його так, щоб поліпшити розв’язок. Другий призначений для вирівнювання положення блоків по горизонталі і вертикалі, що приводить до щільного прилягання їх один до одного.

Висновок

Основна мета схрещування – отримання нових варіантів розв’язків порівняно з тими, що вже існують. Тому обов'язковою умовою є те, що внаслідок схрещування двох батьківських особин повинні виникнути коректні у межах поставленого завдання нащадки. У деяких випадках ця умова

вимагає використання "нестандартних" генетичних операторів. Не варто розцінювати генетичні алгоритми як своєрідну панацею для завдань оптимізації. Існує No Free Lunch теорема, згідно з якою на повній множині завдань не можна виділити найкращий метод оптимізації. Тому з великою вірогідністю генетичні алгоритми покажуть як мінімум не кращі результати порівняно із спеціально розробленими методами.

По-перше. Великою перевагою еволюційних обчислень є можливість використання уніфікованого підходу до вирішення найрізноманітніших проблем. Під час вирішення складних завдань перебору генетичні алгоритми показують прекрасні результати.

По-друге. При реалізації алгоритмів велику увагу необхідно звертати на специфіку мови програмування і нюанси програмування цих алгоритмів. Це може дати певний виграш у часі.

1. Емельянов В. В., Курейчик В. В., Курейчик В. М. Теория и практика эволюционного моделирования. — М.: ФИЗМАТЛИТ, 2003. – 432 с. 2. E. Hadjiconstantinou and N. Christofides, "An exact algorithm for general, orthogonal, two-dimensional knapsack problems", European Journal of Operational Research 83 (1995) 39-56. 3. K.K. Lai and J.W.M. Chan, "Developing a simulated annealing algorithm for the cutting stock problem", Computers & Industrial Engineering 32 (1997) 115-127. 4. R.D. Tsai, E.M. Malstrom and H.D. Meeks, "A two-dimensional palletizing procedure for warehouse loading operations", IIE Transactions 20 (1988) 418–425. 5. Курейчик В.М. Генетические алгоритмы. – Таганрог.:ТРТУ, 1998, 239 с. 6. А.С. Мухачева, А.В. Чиглинцев Генетический алгоритм поиска минимума в задачах двумерного гильотинного раскроя, Информационные технологии. – 2001. – №3. – С. 27–31. 7. Норенков И.П. Эвристики и их комбинации в генетических методах дискретной оптимизации // Информационные технологии. 1999. №1. – С 2–7. 8. Баодин Л., Теория и практика неопределенного программирования. –М.: БИНОМ, Лаборатория знаний 2005, -416 с.: ил. 9. Рутковская Д., Пилинський М., Рутковский Л., Нейронные сети, генетические алгоритмы и нечеткие системы. – М.: Горячая линия-Телеком, 2006. – 452 с. 10. Что такое генетические алгоритмы Тимофей Струнков, PC Week RE, 19/99. 11. Y. Hrytsyshyn, R. Kryvyy, S. Tkatchenko, Genetic Programming For Solving Cutting Problem.//Proceedings of the IXth International Conference on "The Experience of Designing and Application of CAD Systems in Microelectronics" CADSM 2007, Polyana, Ukraine, 2007, pp. 280-282 12. Dmitry Korpyljov, Tatyana Sviridova|, Sergey Tkachenko. Using of genetic algorithms in design of Hybrid Integrated Circuits.//Proceedings of the IXth International Conference on "The Experience of Designing and Application of CAD Systems in Microelectronics" CADSM 2007, Polyana, Ukraine, 2007, pp. 302