УДК 681.324+577.212

Національний Університет
“Києво-Могилянська Академія”

М.М. Глибовець, С.С. Гороховський,
О.В. Краткова

ГІБРИДНИЙ ГЕНЕТИЧНИЙ АЛГОРИТМ ВИРІШЕННЯ ЗАДАЧІ ОПТИМІЗАЦІЇ СТРУКТУРИ ІНТЕГРАЛЬНОЇ СХЕМИ

У роботі описано генетичний алгоритм вирішення задачі оптимізації структури інтег-ральної схеми (Floorplan Optimization) та його реалізацію.

В работе описаны гене-тический алгоритм решения задачи оптимизации струк-тури интегральной схемы (Floorplan Optimization) и его реализация.

A genetic algorithm for the floorplan optimization problem and its implementation are presented in this paper.

Ключові слова: генетичний алгоритм, застосування гене-тичних алгоритмів, оптимізація структури інтегральних схем, Floorplan Optimization.

Вступ

Відомо, що еволюційну метафору можна застосовувати для вирішення оптимізаційних задач чи задач пошуку за умови, що можливі розв’язки задачі можна деяким чином закодувати у рядки символів, коректно визначити оператори вибірки, мутації та схрещування для таких рядків і описати деяку функцію, що оцінюватиме життєздатність такого розв’язку. Оператор схрещування вносить у закодований рядок суттєві зміни, забезпечуючи пошук на всій множині можливих розв’язків, а оператор мутації потроху «налаштовує» оптимальніший розв’язок [1-3].

Дослідження даної теми є безперечно актуальним, адже розробка, аналіз та застосування ефективних і універсальних методів розв’язку задач (алгоритмів) є наразі ключовою задачею комп’ютерних наук, а тема еволюційних алгоритмів ще не достатньо досліджена. Генетичні алгоритми надають можливість швидкої генерації прийнятних розв’язків задач, які неможливо розв’язати іншими аналітичними методами, уникаючи повного перебору і значно скорочуючи часові витрати.

При застосуванні традиційних методів оптимізації і пошуку у разі навіть незначної зміни параметрів середовища всі обчислення доводиться проводити заново. Еволюційний підхід дозволяє аналіз і адаптацію вже створеної популяції до нових умов середовища, чим теж скорочує час роботи алгоритму і реалізує принцип машинної адаптації і навчання.

Генетичні алгоритми застосовні до досить широкого кола задач, які можна звести до глобального чи локального пошуку і оптимізації розв’язку або до симуляції змін у динамічному середовищі. Це досить стійкий і потужний засіб, оскільки, незважаючи на те, що хоча складність задач постійно зростає, а час, відведений на їх розв’язання, все скорочується, системи, засновані на генетичних алгоритмах, успішно справляються з поставленими задачами.

Метою дослідження даної роботи є розробка і демонстрація можливого застосування гібридного генетичного алгоритму для вирішення задачі оптимізації структури інтегральної схеми або знаної задачі оптимізації флорплену (Floorplan Optimization, оптимізація розміщення модулів інтегральної схеми з мінімізацією витрат на площу і зв’язки між деталями) [4].

Генетичні алгоритми

Схематично канонічний ГА можна записати так [5]:

1. Генеруємо початкову популяцію з n хромосом.
2. Обчислюємо для кожної хромосоми її життєздатність.

© ГлибовецьМ.М., ГороховськийС.С., Краткова О.В., 2011

3. За допомогою обраного методу відбору обираємо пару хромосом-батьків.

4. Виконуємо схрещування батьків із ймовірністю p_c. Отримуємо 2 нащадків.

5. Виконуємо мутацію нащадків із ймовірністю p_m.

6. Повторюємо 3-5, доки не буде сформовано нове покоління з n хромосом.

7. Повторюємо 2-6, доки не досягнемо умови зупинки алгоритму.

Згодом уявлення про генетичниі алгоритми значно розширилися і до них наразі відносять і суттєво відмінні від канонічного алгоритми. Наприклад, у техніці генетичного програмування шукані розв'язки мають форму комп'ютерних програм, а оцінка життєздатності полягає у визначенні, чи здатна програма розв'язати деяку обчислювальну задачу. В еволюційному програмуванні, навпаки, структура комп'ютерної програми зафіксована, а еволюціонують параметри. У методі еволюційної стратегії замість двійкових рядків застосовують вектори дійсних чисел, а також часто використовується автоматизація пристосування показників рівня мутації.

Учні Голланда Кеннет Де Йонг та Девід Ґолдберг продовжили його дослідження і зробили чималий внесок в розвиток генетичних алгоритмів. Найвідомішим доробком Ґолдберга є «Генетичні алгоритми в оптимізації пошуку та машинному навчанні» («Genetic algorithms in search optimization and machine learning» (1989)), одна з найбільш цитованих у галузі комп'ютерних наук. Завдяки цим вченим, описані всі генетичні оператори і досліджена поведінка групи тестових функцій і саме алгоритм Ґолдберга отримав назву «генетичний алгоритм».

Від традиційних методів оптимізації генетичні алгоритми відрізняються тим, що [4,5]:

- мають справу не зі значеннями параметрів задачі, а з їх закодованою формою;
- здійснюють пошук рішення не з єдиної вихідної точки, а відштовхуючись від деякої популяції (операції на популяції);
- використовують лише цільову функцію, а не її похідні чи іншу допоміжну інформацію (використання мінімальної інформації про задачу – не потрібна інформація про поведінку функції);
- застосовують ймовірнісні, а не детерміновані правила вибору (рандомізація операцій);
- ГА застосовні до задач, які раніше розв'язувалися лише повним комбінаторним перебором, або лише за допомогою нейронних мереж, а також до раніше нерозв'язних задач.

Генетичні алгоритми дозволяють працювати з широким колом теоретичних і прикладних задач: нейронні мережі, оптимізація багатовимірних функцій (*задача про раціон, транспортна задача* [6], *задача складання розкладу* [1]), в системах підтримки прийняття рішень [6,7], наближених методах (*еволюційні алгоритми* [3]).

Генетичні алгоритми з успіхом застосовують, коли необхідно скомпонувати (розмістити на шматі матеріалу, нерідко обмеженого за розміром) задану множину елементів (інколи навіть враховуючи зв'язки між ними), мінімізуючи витрати на площу (задача крою, дизайн інтер'єрів, планування міста чи ландшафту тощо). У задачі крою одягу може також бути врахована інформація про напрям нитки у деталях. Таким чином, генетичні алгоритми з успіхом застосовують в конструктивному дизайні і створенні креслень майбутніх деталей.

Прикладом задачі компонування також є оптимізація загальної структури інтегральної схеми (floorplan optimization) [4], детальніше розглянута далі.

Інколи генетичний алгоритм спрямований на покращення параметрів деякої характеристики зображення – чіткості контурів, контрастності, освітлення, текстури, кольорової гами тощо. Генетичні технології ефективно допомагають у реконструкції і стилізації фотографій. Наприклад, японськими вченими розроблені ГА для генерації якіснішого чорно-білого зображення із плавними переходами тонів на основі заданої фотографії [8].

Останнім часом, із поширенням комп'ютерних ігор, популярнішою стала генерація тривимірних зображень за допомогою генетичних алгоримтів. Цікавими застосуваннями ГА у цьому напрямку є також генерація відбитків пальців [9], оптимізація відеозображень, генерація стрічкових кодів.

Задачу автоматизації написання музики можна спростити до пошуку деякого розв'язку з неструктурованої множини всіх можливих композицій [10]. Годі зазначати величину цієї множини, беручи до уваги кількість музичних інструментів і можливих на них звуків і акордів. Деякі застосування генетичних лгоритмів в цій царині розглянені в [11].

Інтерактивні генетичні алгоритми є чудовим допоміжним засобом, коли йдеться про креативні рішення та оригінальні задачі. Користувач може, відповідно до своїх смаків, обрати серед численних варіантів, запропонованих відповідною програмою, і не боятися порушення авторських прав на використання ідеї, адже в процесі еволюції варіантів дизайну вирішальну

роль відіграють естетичні уподобання самого користувача [12].

Суттєвим плюсом використання ГА у роботі, пов'язаній з дизайном, є те, що ГА спроможні не тільки на генерацію деякої дизайнерської ідеї, але і на подальше оптимізаційне компонування креслень деталей майбутнього виробу на матеріалі з мінімізацією затрат довжини і/або ширини матеріалу).

Генетичний алгоритм розв'язку задачі Floorplan Optimization

Інтеграція надвеликих інтегральних мікросхем (Very Large-Scale Integration, VLSI) – це процес побудови складних електронних компонентів (мікропроцесорів, чіпів пам'яті), що складаються з великої кількості функціональних компонентів, а тому дизайн VLSI вимагає складних обчислень: перевірка вірності розташування складових чіпа, двовимірне схематичне зображення чіпа, генерація випадків для тестування чіпа, коли він буде готовий тощо. Перша стадія процесу VLSI передбачає виокремлення набору неподільних і, як правило, прямокутних блоків – модулів-клітин (cells). Зрозуміло, що в останнє десятиліття блочна складність нових схем постійно зростає. На другій стадії фіксуються дані про взаємне розташування цих модулів. На третій стадії для модулів обираються різні варіанти імплементації з метою оптимізації загальної площі. У цій роботі основна увага приділена третій стадії VLSI, яку ще називають *оптимізацією загальної топологічної структури (надвеликих) інтегральних мікросхем* (англ., *Floorplan Optimization*). Кінцева вартість чіпа багато у чому визначається величиною його площі, тому знаходження оптимального флорплену його елементів на ранній стадії (до фізичного виготовлення) – це найважливіший етап процесу інтеграції схеми [4]. Знаходження оптимальних флорпленів (структур даних, які представляють взаємне розташування множини об'єктів) – це актуальна царина досліджень комбінаторної оптимізації. Більшість задач про флорплени мають NP-складність і вимагають значних ресурсів пам'яті. Тож найпопулярнішим підходом до таких задач є оптимізаційні евристики для знаходження прийнятних рішень. У цій роботі ми пропонуємо для вирішення такої задачі застосувати гібридний ГА.

Існує дві категорії флорпленів – розрізний (*slicing*) та безрозрізний (*non-slicing*). Перший можна одержати, рекурсивним (горизонтальним чи вертикальним) розрізанням площини на

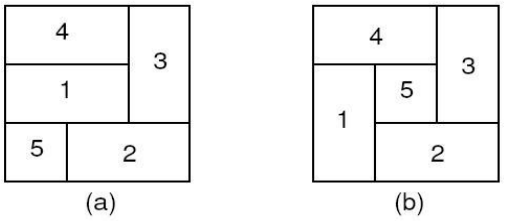

Рис. 1 Приклади простих флорпленів: a) безрозрізний; b) розрізний [4]

менші прямокутники. Другий є більш загальним і може бути зведеним до першого.

На рис. 1 наведені відповідні приклади найпростіших флорпленів з п'яти блоків. Для представлення структури розрізного флорплену в пам'яті комп'ютера використовують бінарні дерева і нормалізовані польські записи, які вказують на порядок «розрізання», а для безрозрізного – різні топологічні представлення, представлення групування інформації (packing representation: O-дерева, B*-дерева), мозаїчні представлення (CBL, Q-послідовність, подвійні бінарні дерева, подвійні бінарні послідовності).

Уточнимо постановку задачі [4]. На вхід оптимізатора подаються такі дані:

- кількість модулів N, які необхідно розмістити на інтегральній схемі;
- для кожного запланованого модуля – визначений список його можливих імплементацій (для кожного модуля їх може бути кілька, з різними довжинами сторін);
- два спеціальні полярні графи (традиційно називаються **G** і **H** графами), які визначають суміжність модулів, відповідно, у вертикальному і горизонтальному напрямках (дуги позначають модулі, а вершини – наявність точок з'єднання).

Ми поширили постановку задачі у [4], поставивши перед собою мету реалізувати повороти модулів на площині схеми для ефективнішого використання матеріалів. При повороті модуля потрібно забезпечити збереження зв'язків між модулями.

Отже, на виході ми маємо отримати схему оптимального флорплену – як саме модулі будуть розміщені на інтегральній схемі з мінімізацією використаної площі.

Однак, якщо ця задача і виглядає досить простою для маленької кількості модулів і їх імплементацій, то у сучасних інтегральних

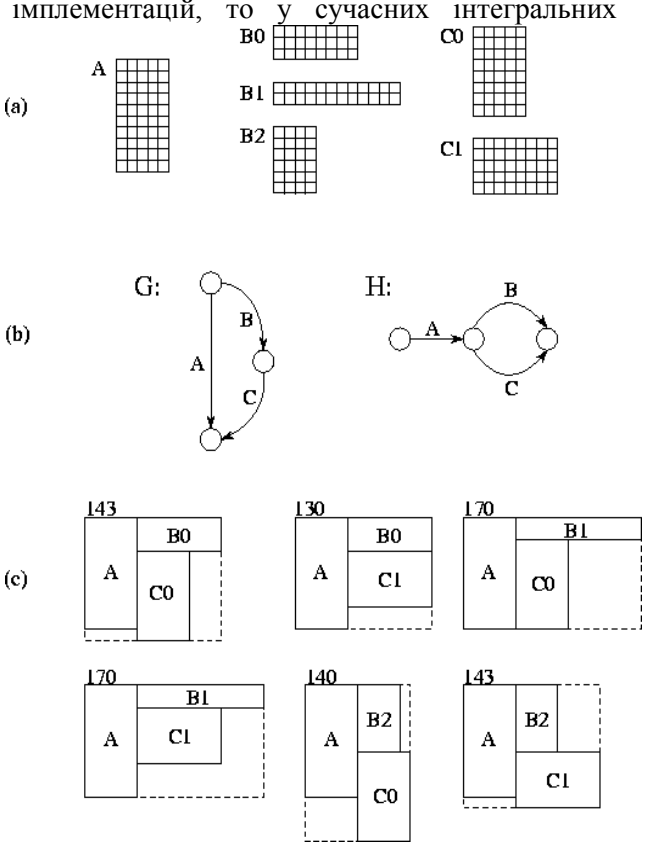

Рис. 2 Приклад Фостера про розв'язання задачі для схеми з трьох модулів [4]

схемах модулів і варіантів їх виконання, звичайно, може бути велика кількість, тому задача значно ускладнюється. Припустимо, що потрібно розмістити *N* модулів, а деякий модуль C_i ($1<i<N$)має $x(C_i)$ варіантів імплементації. Тоді для знаходження кращого розв'язку задачі ми повинні передивитися таку кількість варіантів, які задовольняють умови поставленої задачі:

$$\prod_{i=0}^{N-1} x(c_i)$$

На рисунку 2 наведений графічний приклад розв'язання задачі компонування для трьох модулів відповідно з 1-єю, 3-ьома і 2-ома імплементаціями. Для знаходження найкращого потрібно перебрати 1x3x2=6 варіантів можливих флорпленів.

Оскільки ми розглядаємо задачу із поворотами модулів, повний перебір включатиме таку кількість варіантів:

$$\prod_{i=0}^{N-1} 2x(c_i)$$

Отже, виникає потреба у принципово відмінному від простого перебору методі розв'язання задачі. Зосередимо увагу на особливостях застосування гібридного генетичного алгоритму.

Структура хромосоми. У даному випадку застосування двійкових значень генів хромосоми не є виправданим, тому що таким чином ми не скоротимо перебір, а навпаки його розширимо. Тому використовуватимемо гібридний генетичний алгоритм компонування *N* модулів. Якщо ми закодуємо кожен модуль чотирма послідовними числами (дві координати розташування на схемі, висота і ширина модуля), отримаємо хромосому довжиною 4**N*.

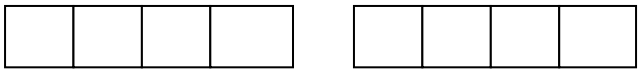

Рис. 3 Схема кодування розв'язку задачі у хромосомі

Особливості ініціалізації популяції. З метою скорочення часових витрат на знаходження потрібних параметрів модулів, ініціалізуємо особини початкової популяції не зовсім випадковим чином. Обиратимемо випадкову імплементацію деякого модуля з відповідної йому множини імплементацій (змінні a_i і c_i на схематичному малюнку), а координати модуля ініціалізуємо випадковим чином. На цьому етапі хромосома може закодувати модулі як без поворотів, так і з ними.

Селекція. Для виконання вибірки хромосом подальшого схрещування ми пропонуємо застосувати селекції двійкового (парного) турніру TournamentSelection [2]. Як найпопулярніший метод селекції, вона чудово підходить для реалізації ГА. Таким чином, зі всієї популяції випадковим чином обираються пари хромосом, а з кожної пари до проміжного масиву відбирається одна, із кращим (меншим) значенням функції пристосованості. Ця процедура повторюється стільки разів, скільки потрібно особин для проміжної популяції. Потім в отриманому проміжному масиві застосовуються генетичні оператори.

Оператори схрещування. У алгоритмі застосовані два гібридних оператори схрещування. Отримані в результаті першого оператора дочірні хромосоми приносять більше розмаїття у популяцію, а у результаті другого – не так радикально еволюціонують (Рисунок 4).

Перший оператор схрещування – оператор одноточкового схрещування *OnePointCrossover* – подібний до оператора схрещування канонічного ГА, за однією тільки відмінністю, що «точка розрізу», за якою обидві батьківські хромосоми, поділяються на два сегменти, обирається випадковим чином між групами генів, що закодовують окремі модулі, а не всередині групи.

Це зумовлено особливостями структури хромосоми у нашому проекті.

За таким само принципом між групами генів відбуваються «розрізи» другого оператора схрещування – оператора двоточкового схрещування. *TwoPointCrossover*. Він «розрізає» хромосоми по краях сегменту генів, відповідального за один і той самий модуль, і за допомогою цього оператора мутації хромосоми «обмінюються» інформацією про імплементацію цього модуля. Схематичне зображення дії обох операторів мутації наведене на рисунку 4 (згори – *OnePointCrossover;* знизу - *TwoPointCrossover*).

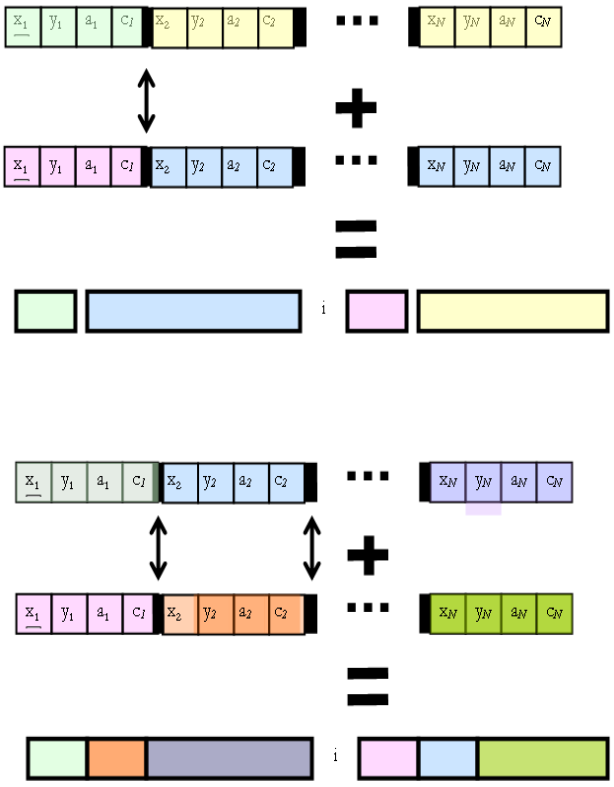

Рис. 4 Оператори схрещування: вгорі – одноточкове схрещування; внизу – двоточкове схрещування

Оператори мутації. У алгоритмі визначено також два гібридні оператори мутації. У їх роботі ми враховуємо такі особливості постановки задачі як скінченність шмату матеріалу і безпосереднє визначення користувачем множини можливих імплементацій конкретного модуля.

Зміст роботи першого визначеного оператора мутації (*MutatePosition*) полягає у мутації гена, відповідального за одну з координат обраного випадковим чином модуля.

Це відбувається за рахунок фактичного збільшення/зменшення параметрів координат. Якщо при збільшенні параметр координати виходить за задані межі площини матеріалу, ми надаємо йому значення 0 і, таким чином, розміщуємо поточний модуль біля краю площини. Цей оператор забезпечує знаходження оптимального місця розташування відповідного модуля на інтегральній схемі.

Зміст другого гібридного оператора мутації *MutateImplementation* полягає у фактичній заміні поточної імплементації модуля на іншу з відповідної множини імплементацій. Ця заміна допомагає нашому оптимізаційному алгоритму уникнути «зациклювання» на деякому локальному екстремумі, а також поступово еволюціонувати у напрямку покращення результату. Також важливим є той факт, що цей оператор мутації із деякою ймовірністю реалізує поворот модуля.

Отже, визначені у проекті гібридні генетичні оператори мутації дозволяють уникнути передчасної збіжності алгоритму, що сприяє підвищенню ефективності його роботи і вносить більше коректності у вихідні результати.

Функція пристосованості. Як вже зазначалося вище, хромосоми в імітації процесу природного відбору потребують певної якісної оцінки їх життєздатності, тому для кожної хромосоми з популяції обчислюється функція пристосованості.

Ця функція показує міру оптимальності розв'язку, тобто чим краще значення функції для хромосоми, тим краще вона як розв'язок задовольняє умовам оптимізаційної задачі.

Було очевидно, що за значення функції пристосованості доведеться взяти величину, яку ми оптимізуємо – загальну площу інтегральної схеми, яка б втілювала даний флорплен.

Однак при втіленні модулів потрібно було врахувати ще й зв'язки між ними, тому додатково була введена перевірка на збереження заданих користувачем зв'язків між модулями із суттєвим збільшенням значення функції пристосованості для хромосоми, яка не відповідає постановці задачі.

Відповідно, краще пристосованою хромосомою вважається та, оцінка функції пристосованості якої менша.

Критерії зупинки роботи алгоритму. Для того, щоб алгоритм працював у межах розумного часу, визначаються критерії зупинки роботи алгоритму.

По-перше, якщо протягом визначеної кількості поколінь не відбувається покращення найліпше пристосованої хромосоми, алгоритм припиняє роботу.

Цей випадок означає, що хромосоми популяції майже однакові і знаходяться у області

деякого екстремуму, а схрещування вже майже ніяк не впливає на популяцію, бо дочірні особини або є абсолютними копіями особин з цієї популяції, або виходять за межі області прийнятних розв'язків і виявляються менш пристосованими, ніж інші особини популяції.

Іншими словами, зупинка відбувається у випадку, коли популяція збіглася – просто знайдений найкращий або близький до нього розв'язок.

По-друге, для того, щоб можна було простежити хід процесу еволюціонування розв'язків задачі, визначений другий критерій призупинення алгоритму. Користувач може переглядати поточний найкращий розв'язок, отриманий після визначеної кількості ітерацій, а також, якщо він вважає його за цілком прийнятний, передчасно зупинити роботу алгоритму на цьому етапі.

Якщо алгоритм зупиняє свою роботу, то на вихід подається хромосома найліпше пристосованої особини (особин) з останньої популяції, інакше цикл починається знову від вибірки хромосом з поточної популяції.

Весь описаний генетичний алгоритм оптимізації загальної топологічної структури інтегральних мікросхем функціонує за схемою канонічного генетичного алгоритму, наведеного раніше.

Висновки

Генетичні алгоритми – потужна і перспективна технологія, яка у своїй роботі використовує імітацію еволюції. Генетичні алгоритми як ефективні метаевристичні алгоритми наближеного розв'язку і оптимізації багатокритерійних задач відіграють вагому роль у сучасній науці.

Постійно створюються нові програмні рішення, що реалізують застосування ГА у різноманітних сферах діяльності людини.

Ці системи полегшують обчислення, уникаючи повного перебору варіантів рішень, значно скорочують часові та машинні витрати на розв'язання поставлених задач.

Генетичні алгоритми надають можливість швидкої генерації прийнятних розв'язків задач, які неможливо розв'язати іншими традиційними аналітичними методами.

Водночас, розвиток царини еволюційних алгоритмів ставить перед наукою повсякчас нові задачі (наприклад, з'явився інтерес до автоматизованої генерації 3-D і 4-D зображень за допомогою ГА). Саме тому інтерес до генетичних алгоритмів невпинно зростає.

З іншого боку, в досліджуваній області є чимало проблем, особливо пов'язаних із розміром динамічних хромосом, тривалістю обчислення цільових функцій, підбором оптимальних параметрів генетичних операторів і кращого оператора селекції для конкретної задачі, застосуванням гібридних генетичних алгоритмів (тобто злиття генетичного алгоритму з традиційними методами, використовуваними у досліджуваній сфері, заснованими на експертних знаннях) тощо.

Універсальних підходів до вирішення цих проблем не існує. Потрібно суттєво досліджувати предметну область застосування і використовувати гібридні алгоритми.

Тому нами розроблено практичний проект реалізації саме гібридного генетичного алгоритму вирішення задачі оптимізації структури інтегральної схеми (також відомої як оптимізація флорплену, Floorplan Optimization).

На вхід задачі подаються модулі і їх можливі реалізації, а також графи просторових відношень між ними (відображають вимоги проектувальника до майбутнього структури схеми). На виході ми отримуємо схематичне креслення флорплену, відповідного до знайденого алгоритмом розв'язку.

Як всі результати роботи технологій наближених методів, цей розв'язок, можливо, не є найкращим для цієї задачі, але він є одним з кращих і прийнятним для практичного застосування.

Список літератури

1. *Глибовець М.М., Медвідь С.О.* Генетические алгоритмы и их использование для решения задачи составления расписания // Кибернетика и системный анализ.– 2003.– № 1.– С. 95–108. Панченко, Т.В. (2007).
2. Генетические алгоритмы: учебно-методическое пособие/ под ред. Ю.Ю. Тарасевича. Астрахань, ИД «Астраханский университет», 2007.
3. *Сиренко, С.* (2007). О классификации приближенных методов комбинаторной оптимизации. International Book Series "Information Science and Computing", Volume "Artificial Intelligence and Decision Making".
4. *Foster, I.* (1995). Designing and Building Parallel Programs. 2.7 Case Study: Floorplan Optimization. Addison-Wesley, 1995.
5. *Рутковская Д., Пилиньский М., Рутковский Л.* (2004). Нейронные сети, генетические алгоритмы и нечеткие системы – М.: Горячая линия - Телеком, 2004 – 452 с.
6. *Курейчик В.М., Родзин С.И.* (2003). Эволюционные вычисления: генетическое и эволюционное программирование. "Новости Искусственного Интеллекта", №5(59), РАИИ, Москва, 2003, стр. 13 – 20.
7. *Францкевич, Г. И., Букарев, А. А., Костюк, В. П.* (2001). Нейросетевые и генетические модели и методы анализа данных, Neuroproject, 2001. http://www.neuroproject.ru/.

8. *Saitoh, F.* (2002). Image Generation with Smooth Gradations from Plural Grey-scaled Images Using Genetic Algorithm. Transactions of the Institute of Electrical Engineers of Japan. Volume 122-C #8, pages 1309-1316.

9. *Cho, U.-K., Hong, J.-H., Cho, S.-B.*. (2007). Automatic Fingerprints Image Generation Using Evolutionary Algorithm. LECTURE NOTES IN COMPUTER SCIENCE, NUMB 4570, 2007, pages 444-453.

10. *Jacob, B. L.(1995).* Composing with genetic algorithms. Proc. International Computer Music Conference (ICMC '95), pp. 452-455. Banff Alberta, September 1995.

11. *Horowitz, D.* (1994). Generating Rhythms with Genetic Algorithms. In Proceedings of the 1994 International Computer Music Conference. Aarhus, Denmark: International Computer Music Association.

12. *Van Largen, P., Wijgaards, N., Brazier, F.* (2001). Towards Designing Creative Artificial Systems. Intelligent Interactive Distributed Systems Group, Computational and Cognitive Models of Creative Design V (Heron Island '01) (Gero, J.S., & Maher, M.L., Eds.), pp. 93–112.

Відомості про авторів

Глибовець Микола Миколайович – доктор фізико-математичних наук, професор, декан факультету інформатики Національного університету "Києво-Могилянська академія", завідувач кафедри інформатики, напрям наукових інтересів – штучний інтелект, інтелектуальні системи, електронна освіта.
E-mail: glib@ukma.kiev.ua

Гороховський Семен Самуїлович – кандидат фізико-математичних наук, доцент кафедри інформатики факультету інформатики, керівник магістерської програми «Інформаційні управляючі системи та технології» Національного університету "Києво-Могилянська академія", напрям наукових інтересів – паралельні та розподілені обчислення, агентні технології.
E-mail: semengor@gmail.com

Краткова Ольга Вадимівна – магістр комп'ютерних наук, аспірант 2-ого року навчання за напрямком "Інформаційні технології, напрям наукових інтересів – генетичні алгоритми, семантичний пошук.
E-mail: zelenenke@gmail.com

Стаття надійшла до редакції 17.02.2011 р.