УДК 519.161

К.т.н., доцент Зорін Ю.М., магістрант Каплан М.Я.

Національний технічний університету України «Київський політехнічний інститут»

ГЕНЕТИЧНИЙ АЛГОРИТМ МІНІМАЛЬНОГО РОЗФАРБУВАННЯ ГРАФА

Вступ

Пошук мінімального розфарбування графа – складна комбінаторна задача з класу NP-складних. Застосовується вона для складання розкладів, розподілу регістрів та частот тощо. Для її ефективного розв’язання були розроблені мета-евристичні методи локального пошуку – пошук із заборонами, імітації відпалу та мурашина колонія [1, 2].

Постановка задачі

Метою роботи є розробка генетичних операторів – кроссовера та мутації для генетичного алгоритму мінімального розфарбування графа. Запропоновані оператори, на відміну від традиційних, дозволяють отримувати розв’язок задачі у випадках складних графів з високою щільністю ребер.

Кодування геному та оцінка міри пристосованості

Для графа $G=(V,E)$ геном кодується як $s=<c(v_1),c(v_2),...c(v_n)>$, де n – кількість вершин $c(v_i)\in 1..k$ – колір вершини v_i, k – кількість кольорів. Це кодування є інтуїтивно очевидним і дає можливість подати дані у вигляді масиву пар (номер вершини - колір вершини).

Для кожного індивіда s функція пристосованості $f(s)$ є величина обернена до кількості конфліктів

$$f(s)=\frac{1}{\sum\limits_{q(v_i,v_j)\in E} q(v_i,v_j)}\text{ , де } q(v_i,v_j)=\begin{cases}1, c(v_i)=c(v_j),\\ 0, c(v_j)\neq c(v_i).\end{cases}$$

Верхня оцінка хроматичного числа та початкова популяція

Задача мінімального розфарбування графа може бути зведена до виконання *k*-розфарбування графа і, якщо це вдалося, *k* зменшується на одиницю і процес повторюється поки це є можливим, або буде досягнута максимальна кількість ітерацій.

Саме тому важливо отримати верхню границю хроматичного числа $\chi(G)$ заданого графу. За теоремою Брука $\chi(G) \le \Delta(G)+1$, де $\Delta(G)$ – максимальний степінь вершин графу G. У [3] дається інша оцінка, яка в деяких випадках дозволяє отримати кращі результати: $\chi(G) \le \frac{s}{s+1}(\Delta(G)+2)$, де s – максимальна кількість вершин з однаковим степенем, не меншим за $(\Delta(G)+2)/2$.

Для формування початкової популяції використовується модифікований жадібний алгоритм [4], доповнений декількома ітераціями локального пошуку із заборонами, що дає добру рандомізацію початкової популяції та дещо нижчу кількість конфліктів, ніж при суто випадковому формуванні.

Генетичні оператори

В задачах комбінаторної оптимізації при використанні генетичних алгоритмів (ГА) доцільно використовувати спеціалізовані («змістовні») кроссовери, що спираються на структуру задачі (графа).

Кроссовер, запропонований у роботі, базується на принципі об’єднання безконфліктних незалежних підмножин вершин графа. Об’єднуються підмножини J_{p_1,c_1} і J_{p_2,c_2}, якщо J_{p_1,c_1} – найбільша безконфліктна підмножина кольору c_1 індивіда p_1 і J_{p_2,c_2} таке, що $\forall c_2 \in [1..k] | J_{p_1,c_1} \cap J_{p_2,c_2}$ – максимальне. Результат визначається такими об’єднаннями для кожного кольору, що виконуються з деякою, достатньо високою ймовірністю. Вершини, що не потрапили в жодне об’єднання фарбуються в колір першого індивіда. Крім того проводиться двоточкове схрещування незалежних підмножин з ймовірністю, що також є параметром операції кроссоверу. Це додає алгоритму випадковості і допомагає уникати дочасної збіжності популяції.

Crossover():

визначити безконфліктні підмножини(U_1, U_2) для кожного кольору кожного індивіду;

```
if(one_point_rate < rand()):
    провести двоточкове схрещування U1 та U2;
for(для кожної підмножини uc з U1):
    if(union_rate < rand()):
        знайти в U2 таку підмножину u'c1, що
        uc ∩ u'c1 – максимальне;
        пофарбувати всі вершини результуючого індивіда,
        які є в u та u' в колір c;
```

Операція мутації відіграє важливу роль у ГА, адже вона дозволяє уникати дочасної збіжності популяції та поліпшувати міру пристосованості індивіда. У задачі знаходження мінімального розфарбування графа в ролі мутації як правило застосовують мета-евристичні алгоритми локального пошуку [2, 4]. У цій роботі запропоновано новий метод мутації (мутація мінімальної інтенсивності - MMI). Інтенсивність кольору $c(v_i)$ відносно вершини v_i – це кількість вершин цього кольору, які з'єднані з v_i.

$$I_{v_i}(c) = \sum_{j:e_{ij}\in E} K(v_j), \quad K(v_j) = \begin{cases} 1, c(v_j) = c(v_i), \\ 0, c(v_j) \neq c(v_i). \end{cases}$$

Принцип роботи алгоритму виходить з формулювання задачі розфарбування графа за допомогою ГА: на кожному кроці він намагається зменшити кількість конфліктів певної вершини графа. Алгоритм послідовно розглядає всі конфліктні вершини графа і відносно кожної з них знаходить інтенсивність кольорів. Після цього дана вершина фарбується в колір з найменшою інтенсивністю, що може зменшити кількість конфліктів. Якщо деякий колір має інтенсивність 0 (жодна сусідня вершина не пофарбована в якийсь колір) – вершина стає безконфліктною.

```
MinIntensityMutation():
    while(існують нерозглянуті вершини):
        if(вершина має конфлікти):
            знайти сусідні вершини, які з'єднані з поточною;
            визначити інтенсивність кожного з кольорів, в які
            пофарбовані сусіди;
            пофарбувати поточну вершину у колір з найменшою
            інтенсивністю;
```

Варто зазначити, що цей алгоритм принципово відрізняється від DSATUR [4], який розглядає вершини з найбільшою кількістю різнокольорових сусідів.

Результати

Для оцінки якості запропонованого ГА було використано складні графи (500 та 1000 вершин) з набору тестів DIMACS. Результати тестування наведені в табл. 1.

Таблиця 1. Результати тестування ГА

Граф	Хроматичне число $\chi(G)$	К-сть запусків	Найгірше k
DSJC500.1.col	12	3	14
DSJC500.5.col	48	5	50
DSJC500.9.col	126	8	130
DSJC1000.1.col	20	3	23
DSJC1000.5.col	83	4	85
DSJC1000.9.col	224	9	231

Оптимальними параметрами ГА виявилися: розмір популяції 100-120, ймовірність мутації – 0.05, кроссовер - ймовірності 0.13 та 0.85 для схрещування та об’єднання відповідно.

При порівнянні з [1] запропонований алгоритм дає аналогічні результати, але за рахунок використання двох параметрів кроссоверу може працювати швидше й частіше уникати передчасної збіжності. Запропонований оператор мутації MMI має значно меншу складність, ніж пошук з заборонами.

Запропонований алгоритм може бути поліпшено якісно за пошуковою силою за рахунок введення додаткових методів мутації, кожен з яких виконується з певною ймовірністю та подальші модифікації запропонованого кроссоверу.

Література

1. *R. Dorne, Jin-Kao Hao.* A New Genetic Local Search Algorithm for Graph Coloring. Parallel Problem Solving. – Springer Berlin, 1998. – P. 745-754.
2. *C. Fleurent, J.A. Ferland*. Genetic and hybrid algorithms for graph coloring // Annals of Operations Research, 1995. – № 63 - P. 437-463.
3. *L. Stacho.* A Note on Upper Bound for Chromatic Number of a Graph // Acta Math. Univ. Comenianae Vol. LXXI, 1, 2002. – P. 97-99.
4. *P.Galinier, Jin-Kao Hao.* Hybrid Evolutionary Algorithms for Graph Coloring // Journal of Combinatorial Optimization, 1999. - №3, P. 379–397.
5. *Д. Рутковская, М. Пилинський*. Нейронные сети, генетические алгоритмы и нечеткие системы. – М.: видав. PWN, 2006. – С.124-169.