УДК 519.174.7

АНАЛИЗ КАЧЕСТВА РЕШЕНИЙ СЕМЕЙСТВА АЛГОРИТМОВ ЕРШОВА-КОЖУХИНА ДЛЯ РАСКРАСКИ ГРАФА

Советов П.Н.

МИРЭА - Российский технологический университет, 119454, Россия, г. Москва, проспект Вернадского, 78, e-mail: sovetov@mirea.ru

К задаче раскраски графа можно свести различные задачи планирования и размещения ресурсов. В частности, раскраска графов используется в области высокоуровневого синтеза цифровых схем. Точное решение для раскраски графа не всегда практически достижимо. По этой причине интерес представляют эвристические алгоритмы раскраски графа, позволяющие получить допустимое решение с полиноминальной вычислительной сложностью. Такие алгоритмы могут использоваться, в частности, в задаче быстрого поиска в пространстве архитектурных вариантов при высокоуровневом синтезе вычислительной системы. В этой связи интерес представляет семейство эвристических алгоритмов Ершова-Кожухина, о котором до сих пор отсутствовали упоминания в современной литературе по анализу алгоритмов раскраски графа. В статье приводится краткое описание и псевдокод алгоритмов Ершова-Кожухина, а также приводится их экспериментальная оценка в сравнении с алгоритмами Welsh-Powell, RLF и DSATUR. Полученная оценка свидетельствует о том, что алгоритмы Ершова-Кожухина имеют преимущество в качестве формируемых решений по сравнению с популярными эвристическими алгоритмами раскраски графа.

Ключевые слова: раскраска графа; эвристический алгоритм; высокоуровневое проектирование; поиск в пространстве архитектурных вариантов, алгоритм Ершова-Кожухина.

ANALYSIS OF SOLUTION QUALITY OF THE FAMILY OF YERSHOV-KOZHUKHIN ALGORITHMS FOR GRAPH COLORING

Sovietov P.N.

MIREA - Russian Technological University», 119454, Moscow, 78 Vernadskogo Avenue, Russia, e-mail: sovetov@mirea.ru

Various planning and resource allocation problems can be reduced to the graph coloring problem. In particular, graph coloring is used in high-level synthesis of digital circuits. The exact solution for graph coloring is not always practically achievable. For this reason, heuristic graph coloring algorithms, which allow to obtain a feasible solution with polynomial computational complexity, are of interest. Such algorithms can be used, in particular, in the problem of fast design space exploration in high-level synthesis of digital system. In this regard, the Ershov-Kozhukhin family of heuristic algorithms, which has been so far unmentioned in the modern literature on analysis of graph coloring algorithms, is of interest. A brief description and pseudocode of the Ershov-Kozhukhin algorithms is given and their experimental evaluation in comparison to the Welsh-Powell, RLF and DSATUR algorithms is presented in the paper. The evaluation shows that the Ershov-Kozhukhin algorithms have an advantage in the quality of the generated solutions compared to popular heuristic graph coloring algorithms.

Keywords: graph coloring; heuristic algorithm; high-level synthesis, design space exploration, Ershov-Kozhukhin algorithm.

Введение

Задача раскраски графа заключается в назначении номеров («красок») вершинам графа таким образом, чтобы все пары соседних вершин имели различные цвета, а общее число красок оказалось минимальным. К этой задаче можно свести различные задачи планирования и размещения ресурсов. В частности, с использованием раскраски графа решаются различные задачи в области высокоуровневого синтеза цифровых схем. Среди примеров можно привести задачу синтеза распределенных блоков памяти с параллельным доступом [1], а также задачу размещения регистров в тракте данных [2]. Следует отметить, что задача распределения регистров в компиляторах здесь не упоминается, поскольку требует использования специализированных алгоритмов раскраски графа с помощью заранее заданного числа красок и с учетом выгрузки величин в основную память при недостаточном числе регистров [3].

Задача раскраски графа является NP-трудной [4] и поэтому использование точных алгоритмов для решения этой задачи не всегда практически осуществимо. В области высокоуровневого синтеза цифровых схем одним из важных начальных этапов является исследование пространства проектирования [5] с целью быстрого получения и сравнения между собой результатов синтеза с различными архитектурными параметрами. Существуют методы, позволяющие с полиномиальной вычислительной сложностью оптимальным образом решить задачу размещения регистров в тракте данных, но только для хордальных графов программ [6]. В общем же случае интерес представляют эвристические алгоритмы, позволяющие за полиномиальное время получить раскраску для произвольного графа с допустимым качеством.

К популярным эвристическим алгоритмам раскраски графа относятся Welsh-Powell, Recursive Largest First (RLF) и DSATUR, характеристики которых детально изучены в литературе [4,7]. Вместе с тем, в современных работах по анализу эвристических алгоритмов раскраски графа отсутствуют упоминания о семействе алгоритмов Ершова-Кожухина [8,9]. Анализ качества решений этих алгоритмов представляет интерес из-за их глубокой математической проработанности, позволяющей ожидать результатов раскраски высокого качества, а также с точки зрения использования дополнительных ограничений в процессе нахождения решений. В частности, алгоритмы Ершова-Кожухина применяются для экономии памяти не только отдельных величин, но и для массивов различных длин с совмещением нескольких массивов в одном массиве достаточного размера [9].

Далее в статье приведено краткое описание и псевдокод семейства алгоритмов Ершова-Кожухина. На наборе случайных графов получена экспериментальная оценка этих алгоритмов в сравнении с тройкой известных алгоритмов Welsh-Powell, RLF и DSATUR. Полученные данные демонстрируют практическую применимость алгоритмов Ершова-Кожухина и их преимущество в качестве формируемых решений по сравнению с популярными эвристическими алгоритмами раскраски графа.

Описание анализируемых алгоритмов

Семейство алгоритмов Ершова-Кожухина основано на доказанном в [8] утверждении: существует порядок проведения последовательных объединений («склеиваний») в неполном графе пар вершин, имеющих минимальное взаимное расстояние равное 2, дающий минимальную раскраску. Склеиваемые вершины, при этом, получают один и тот же цвет. Для определения пары склеиваемых вершин могут использоваться различные эвристики. Выбором конкретной эвристики отличаются рассматриваемые далее алгоритмы E1, E2 и E3.

В качестве входного представления графа в этих алгоритмах используется таблица списков смежности. На входной граф накладывается ограничение – он должен быть связным. По этой причине далее предполагается, что предварительным этапом раскраски графа является нахождение компонент связности, к каждой из которых затем применяется один из рассматриваемых алгоритмов.

На рис. 1 представлен псевдокод алгоритма E1. Этот алгоритм основан на последовательном просмотре всех вершин графа (строки 5–11). На каждой итерации выбирается первая доступная вершина m из окрестности на минимальном расстоянии 2 (R_2, строка 8), относящейся к текущей просматриваемой вершине n. Найденная пара вершин склеивается (строка 10), то есть к вершине n добавляются смежные с m вершины и всюду в графе связи с m заменяются связями с n. Алгоритм E1 имеет вычислительную сложность $O(n^2)$.

```
1:  function E1_COLORING(G)
2:      G′ ← copy of G
3:      C ← ∅
4:      current_color ← 0
5:      for each n ∈ G′ do
6:          if n ∉ C then
7:              C[n] ← current_color
8:              while m ∈ R₂(G′, n) do
9:                  C[m] ← current_color
10:                 G′[n] ← glue_nodes(G′, n, m)
11:             current_color ← current_color + 1
12:     return C
```

Рис. 1 - Псевдокод алгоритма E1

На рис. 2 представлен псевдокод алгоритма E2. Единственным изменением по сравнению с E1 в этом алгоритме является целенаправленный выбор вершины из R_2. Вершина m (строка 9) имеет наибольшее число разделяющих вершин с вершиной n. Алгоритм E2 имеет вычислительную сложность $O(n^2)$.

```
1:  function E2_COLORING(G)
2:      G′ ← copy of G
3:      C ← ∅
4:      current_color ← 0
5:      for each n ∈ G′ do
6:          if n ∉ C then
7:              C[n] ← current_color
8:              while |R₂(G′, n)| ≠ 0 do
9:                  m ← argmax_{x∈R₂(G′,n)} |G′[n] ∩ G′[x]|
10:                 C[m] ← current_color
11:                 G′[n] ← glue_nodes(G′, n, m)
12:             current_color ← current_color + 1
13:     return C
```

Рис. 2 - Псевдокод алгоритма E2

Псевдокод алгоритма E3 показан на рис. 3. Очередная итерация этого алгоритма состоит в нахождении такой пары вершин, отстоящих друг от друга на расстоянии 2, для которых число разделяющих их вершин будет наибольшим (строка 5), что позволяет на каждом шаге удалять из графа наибольшее число ребер. Помимо склеивания найденной пары вершин (строка 7) осуществляется также пополнение множества вершин, имеющих общий цвет (строка 6). На завершающем этапе эти объединенные вершины нумеруются цветом своего множества (строки 9–10). Алгоритм E3 имеет повышенную вычислительную сложность – $O(n^3)$. В [9] дан псевдокод оригинальной реализации этого алгоритма – с использованием матрицы смежности.

1: **function** E3_COLORING(G)
2: $\quad G' \leftarrow$ copy of G
3: $\quad C \leftarrow \{(n, \varnothing) : n \in G'\}$
4: $\quad$ **while** $i \in G' \wedge j \in G' \wedge j \in R_2(G', i)$ **do**
5: $\quad\quad i, j \leftarrow \underset{i \in G', j \in G', j \in R_2(G', i)}{\arg\max} |G'[i] \cap G'[j]|$
6: $\quad\quad C[i] \leftarrow C[i] \cup \{j\} \cup C[j]$
7: $\quad\quad G'[i] \leftarrow \text{glue\_nodes}(G', i, j)$
8: $\quad\quad$ delete $C[j], G'[j]$
9: $\quad C' \leftarrow \{\{n\} \cup C[n] : n \in C\}$
10: $\quad$ **return** $\{(n, i) : n \in C_i, C_i \in C', i \in \{0, \dots, |C'| - 1\}\}$

Рис. 3 - Псевдокод алгоритма E3

Экспериментальная оценка качества решений анализируемых алгоритмов

Оценка качества решений алгоритмов E1–E3 реализована на наборе случайных графов. На рис. 4 показан псевдокод используемого алгоритма генерации случайного графа.

function GET_RANDOM_GRAPH(min_nodes, max_nodes)
$\quad G \leftarrow \varnothing$
$\quad num\_nodes \leftarrow$ random integer $\in \{min\_nodes, \dots, max\_nodes\}$
$\quad num\_edges \leftarrow$ random integer $\in \{1, \dots, n(n-1)/2\}$
$\quad$ **for** $i \leftarrow 1$ to num_edges **do**
$\quad\quad n_1 \leftarrow$ random integer $\in \{0, \dots, num\_nodes - 1\}$
$\quad\quad n_2 \leftarrow$ random integer $\in \{0, \dots, num\_nodes - 1\} \setminus \{n_1\}$
$\quad\quad G[n_1] \leftarrow G[n_1] \cup \{n_2\}$
$\quad\quad G[n_2] \leftarrow G[n_2] \cup \{n_1\}$
$\quad$ **return** G

Рис. 4 - Псевдокод алгоритма генерации случайного графа

Для сравнения с алгоритмами E1–E3 выбраны реализации жадных алгоритмов Welsh-Powell, RLF и DSATUR из библиотеки NetworkX 3.1. Используется их комбинированный вариант, называемый далее алгоритмом NX, в котором количество получаемых красок c определяется как:

$$c = \min(c_{\text{WP}}, c_{\text{RLF}}, c_{\text{DSATUR}})$$

На рис. 5 демонстрируются результаты сравнения эвристических алгоритмов раскраски графа на наборе из 5000 случайных графов. Случайные графы генерируются с числом вершин в диапазоне $[50,150]$. Из графика видно, в частности, что алгоритм E1 значительно уступает алгоритму NX в качестве полученных раскрасок графов.

На рис. 6 представлены результаты сравнения алгоритмов E2 и NX. По результатам можно сделать вывод, что E2 превосходит NX в качестве решений. Это превосходство достигается в 41% случаев, при этом NX формирует лучшие чем E2 решения лишь в 10% случаев.

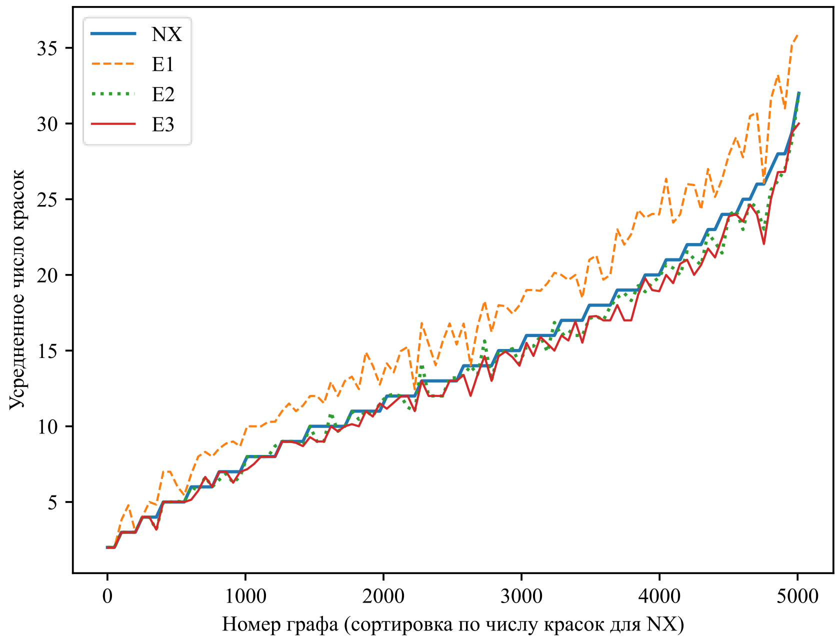

Рис. 5 - Сравнение работы эвристических алгоритмов раскраски графа

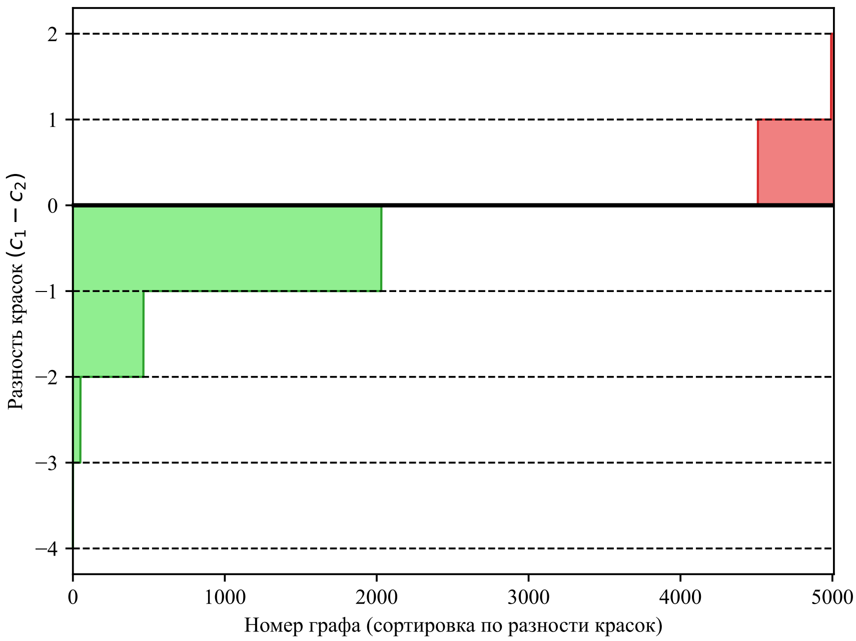

Рис. 6 - Сравнение работы алгоритма E2 (c_1) с алгоритмом NX (c_2)

На рис. 7 показаны результаты сравнения алгоритма E3 с NX. С помощью E3 получены лучшие решения в 54% случаев, а с помощью NX – только в 5% случаев. Это позволяет считать алгоритм E3 формирующим наиболее качественные решения среди рассмотренных алгоритмов.

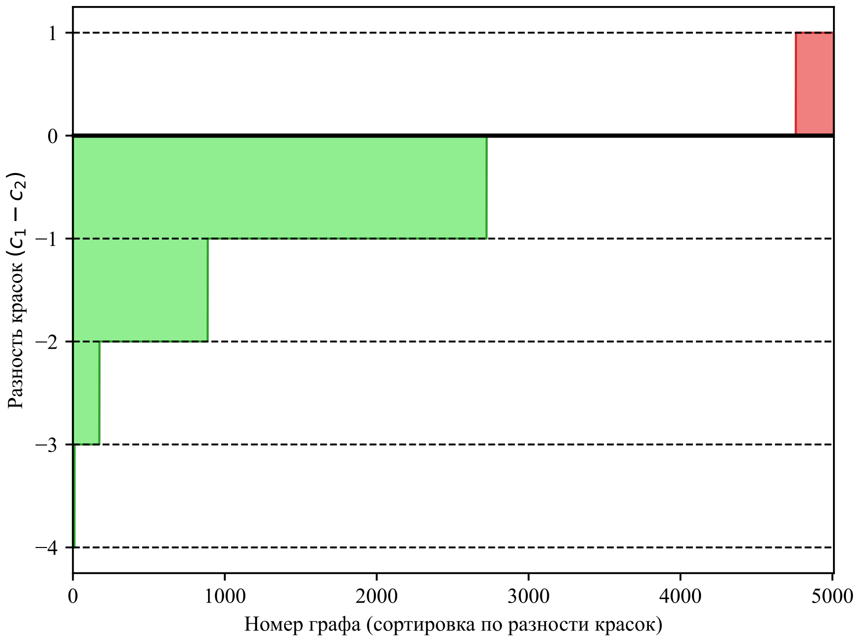

Рис. 7 - Сравнение работы алгоритма E3 (c_1) с алгоритмом NX (c_2)

Заключение

Приведено краткое описание и псевдокод семейства эвристических алгоритмов Ершова-Кожухина для раскраски графа. Получены экспериментальные оценки этих алгоритмов в сравнении с популярными эвристическими алгоритмами Welsh-Powell, RLF и DSATUR.

На основе экспериментальной оценки, полученной на наборе случайных графов, можно констатировать, что алгоритм E3 производит наиболее качественные результаты среди проанализированных алгоритмов, но при этом имеет наибольшую вычислительную сложность $O(n^3)$. Алгоритм E2 является более быстродействующим алгоритмом раскраски графа и не сильно уступает E3 в качестве производимых решений. Это позволяет сделать вывод о перспективности использования E2 и E3 в таких задачах, как быстрое исследование пространства архитектурных вариантов в области высокоуровневого синтеза цифровых схем.

Список литературы

1. Escobedo J., Lin M. Extracting data parallelism in non-stencil kernel computing by optimally coloring folded memory conflict graph / Proceedings of the 55th annual design automation conference. — ACM, 2018.

2. Castellana V. G., Tumeo A., Ferrandi F. High-level synthesis of parallel specifications coupling static and dynamic controllers / 2021 IEEE international parallel and distributed processing symposium (IPDPS). — IEEE, 2021.

3. VenkataKeerthy S., Jain S., Kundu A., Aggarwal R., Cohen A., Upadrasta R. RL4ReAl: Reinforcement learning for register allocation / Proceedings of the 32nd ACM SIGPLAN international conference on compiler construction. — ACM, 2023.

4. Lewis R. M. R. Guide to graph colouring. — Springer International Publishing, 2021.

5. Schafer B. C., Wang Z. High-level synthesis design space exploration: Past, present, and future // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. — Institute of Electrical; Electronics Engineers (IEEE), 2020. — Vol. 39, no. 10. — P. 2628–2639.

6. Canesche M., Ferreira R., Nacif J. A., Pereira F. M. Q. A polynomial time exact solution to the bit-aware register binding problem / Proceedings of the 31st ACM SIGPLAN international conference on compiler construction. — ACM, 2022.

7. Postigo J., Soto-Begazo J., Fiorela V. R., Picha G. M., Flores-Quispe R., Velazco-Paredes Y. Comparative analysis of the main graph coloring algorithms / 2021 IEEE colombian conference on communications and computing (COLCOM). — IEEE, 2021.

8. Ершов А. П., Кожухин Г. И. Об оценках хроматического числа связных графов / Доклады академии наук. — Российская академия наук, 1962. — Vol. 142. — P. 270–273.

9. Ершов А. П. Введение в теоретическое программирование. — Наука, 1977.

References

1. Escobedo J., Lin M. Extracting data parallelism in non-stencil kernel computing by optimally coloring folded memory conflict graph / Proceedings of the 55th annual design automation conference. — ACM, 2018.

2. Castellana V. G., Tumeo A., Ferrandi F. High-level synthesis of parallel specifications coupling static and dynamic controllers / 2021 IEEE international parallel and distributed processing symposium (IPDPS). — IEEE, 2021.

3. VenkataKeerthy S., Jain S., Kundu A., Aggarwal R., Cohen A., Upadrasta R. RL4ReAl: Reinforcement learning for register allocation / Proceedings of the 32nd ACM SIGPLAN international conference on compiler construction. — ACM, 2023.

4. Lewis R. M. R. Guide to graph colouring. — Springer International Publishing, 2021.

5. Schafer B. C., Wang Z. High-level synthesis design space exploration: Past, present, and future // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. — Institute of Electrical; Electronics Engineers (IEEE), 2020. — Vol. 39, no. 10. — P. 2628–2639.

6. Canesche M., Ferreira R., Nacif J. A., Pereira F. M. Q. A polynomial time exact solution to the bit-aware register binding problem / Proceedings of the 31st ACM SIGPLAN international conference on compiler construction. — ACM, 2022.

7. Postigo J., Soto-Begazo J., Fiorela V. R., Picha G. M., Flores-Quispe R., Velazco-Paredes Y. Comparative analysis of the main graph coloring algorithms / 2021 IEEE colombian conference on communications and computing (COLCOM). — IEEE, 2021.

8. Ershov A. P., Kozhuxin G. I. Ob ocenkax xromaticheskogo chisla svyazny`x grafov / Doklady` akademii nauk. — Rossijskaya akademiya nauk, 1962. — Vol. 142. — P. 270–273 (in Russian).

9. Ershov A. P. Vvedenie v teoreticheskoe programmirovanie. — Nauka, 1977 (in Russian).