электронный научно-технический журнал
ИНЖЕНЕРНЫЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51036. ISSN 2307-0595

СРАВНИТЕЛЬНЫЙ АНАЛИЗ АЛГОРИТМОВ РАСКРАСКИ ГРАФА

02, февраль 2017

Зинченко Л. А.[1], Резчикова Е. В.[1], Кобецкий В. А.[1,*], Суров П. В.[1]

УДК: 519.174.7

[1]Россия, МГТУ им. Н.Э. Баумана
*rk62ck@mail.ru

Введение

Разнообразные задачи, возникающие при планировании производства, составлении графиков осмотра, хранении и транспортировке товаров, в молекулярной биологии, нано-инженерии и др., могут быть представлены как задачи теории графов. В частности, ДНК может быть представлена с помощью графа и при секвенировании генома человека были использованы алгоритмы поиска гамильтонова цикла.

Практически важной задачей также является раскраска графа. Для решения таких задач необходимо разбить множество всех вершин графа в объединение непустых непересекающихся подмножеств таким образом, чтобы вершины из одного и того же подмножества были попарно не смежными, а число таких подмножеств – минимально возможным [1 - 3].

Задача нахождения минимально возможной раскраски произвольного графа явилась предметом многих исследований в конце XIX и в последующих столетиях. Все известные точные алгоритмы, которые находят минимальную $\chi(G)$ – раскраску для любого графа, требуют, в лучшем случае, экспоненциального числа шагов [4]. Поэтому для решения подобных задач часто используются приближенные методы, в частности алгоритм прямого неявного перебора. В статье приводятся результаты сравнения этого алгоритма с точным алгоритмом полного перебора.

1. Основные понятия и определения

Раскраской вершин графа называется назначение цветов его вершинам. Обычно цвета – это числа 1, 2, …, *k*. Раскраска называется правильной, если каждый цветной класс является независимым множеством (независимое множество вершин графа – это любое множество попарно не смежных вершин, т.е. множество вершин, порождающее пустой подграф).

Задача о раскраске состоит в нахождении правильной раскраски данного графа *G* с наименьшим числом цветов. Это число называется *хроматическим числом* $\chi(G)$ графа *G* – минимальное число цветов, требующееся для раскраски *G*.

1.1. Алгоритм полного перебора

Полный перебор (или метод «грубой силы», англ. *brute force*) - метод решения математических задач. Относится к классу методов поиска решения путем перебора всевозможных вариантов (метод проб и ошибок (МПО)). Сложность полного перебора зависит от количества всех возможных решений задачи. Если пространство решений очень велико, то полный перебор может не дать результатов в течение нескольких лет или даже столетий.

Рассмотрим работу алгоритма на примере графа (рис. 1). Пусть цветам будут соответствовать коды: 0 – черный, 1 – красный, 2 – зеленый, 3 – синий.

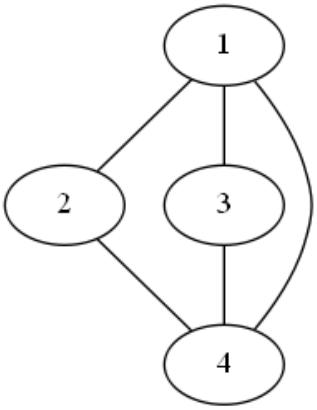


Рис. 1. Граф для раскраски

Алгоритм будет перебирать комбинации цветов до тех пор, пока не будет пройдена проверка, что никакие смежные вершины не имеют одинакового цвета, и граф не будет правильно раскрашен.

Перебор выглядит следующим образом:

0000 – проверка не пройдена;

0001 – проверка не пройдена;

0002 – проверка не пройдена;

0003 – проверка не пройдена;

0010 – проверка не пройдена;

……

0112 – проверка пройдена.

Результат раскраски представлен на рис. 2, где вершина 1 имеет цвет 0, вершины 2 и 3 – цвет 1, вершина 4 – цвет 2.

1.2. Алгоритм неявного перебора

В методах неявного перебора делается попытка так организовать перебор, используя свойства рассматриваемой задачи, чтобы отбросить часть допустимых решений.

Алгоритм *неявного перебора* (англ. *implicit enumeration*) для раскраски вершин графа – это метод, который улучшает жадную раскраску графа путем добавления, так называемого, шага возвращения.

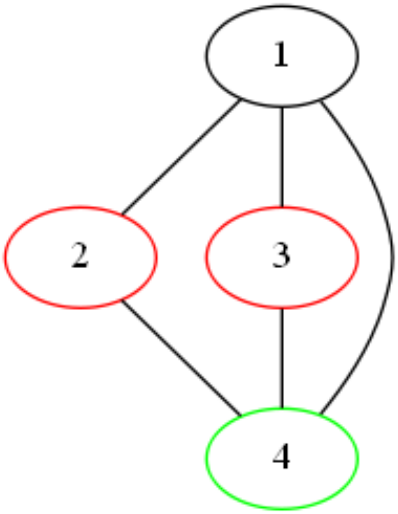


Рис. 2. Раскраска полным перебором

В *жадном алгоритме* (англ. *greedy algorithm*) всегда делается выбор, который кажется самым лучшим в данный момент - т.е. производится локально оптимальный выбор в надежде, что он приведет к оптимальному решению глобальной задачи. Он задает начальную раскраску – *первая фаза* алгоритма.

Во *второй фазе* рассматриваются вершины, смежные с первой вершиной максимального цвета q, но по порядку расположения, находящиеся раньше неё (т.е. делаем шаг возвращения из вершины, требующей максимальный цвет). Предположим, что первая найденная вершина имеет цвет k, тогда пытаемся перекрасить её в новый цвет k' такой, что $q > k' > k+1$.

Если это невозможно, делаем из этой вершины шаг возвращения (с учетом того, что максимальный цвет q не изменился) и продолжаем перебирать вершины, смежные с первой вершиной максимального цвета или с этой – новой вершиной, из которой мы, только что, сделали шаг возвращения, пока не дойдем до первой вершины, при заданном упорядочивании.

Если же встретилась вершина, которую можно перекрасить, то перекрашиваем все следующие за ней вершины, выбирая минимально возможный цвет. При этом, если снова встречается вершина, требующая максимальный цвет q, из неё делается шаг возвращения.

По окончанию второй фазы, когда дошли до первой вершины, проверяем был ли уменьшен максимальный цвет, если да, переходим к началу второй фазы и пробуем улучшить раскраску еще раз.

Проиллюстрируем работу алгоритма для графа, изображенного на рис.1.

Первой фазе, задающей раскраску жадным методом, соответствует рис.3:

Вершина $V1$ окрашена в цвет 1. $V2$ смежна $V1$ - значит, красим в минимально возможный цвет 2. $V3$ также смежна только $V1$ – цвет 2. $V4$ смежна $V1$ и $V2$ - следовательно, минимально возможный для неё цвет 3.

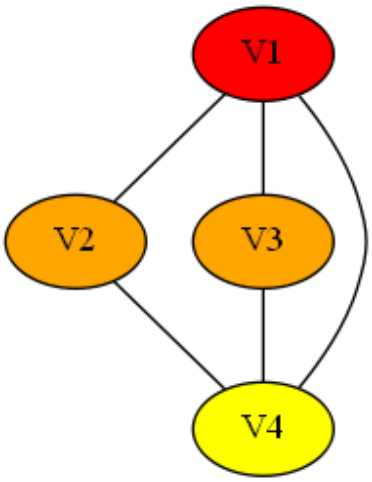


Рис. 3. Раскраска неявным перебором

Далее пытаемся улучшить раскраску по правилу второй фазы. Делаем шаг возвращения из вершины *V*4.

1. Проверяем меньшую по индексу *V*3 – нельзя перекрасить, делаем из нее шаг возвращения; следующая вершина, смежная с *V*4, – *V*2.

2. Проверяем *V*2 – также нельзя перекрасить; переходим к последней *V*1.

3. Дошли до *V*1. Раскраску улучшить не удалось, завершаем работу.

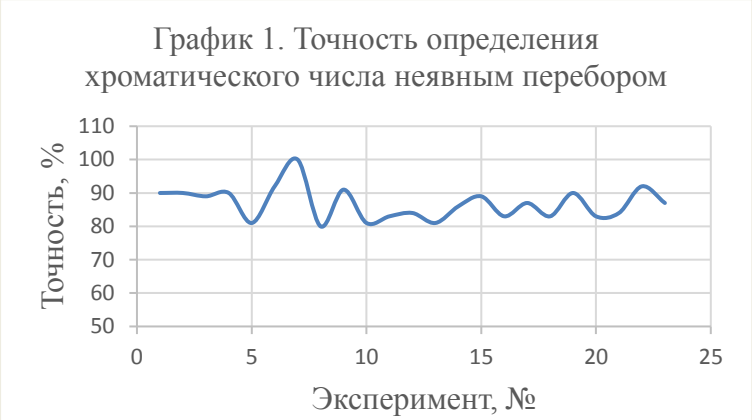
1.3. Сравнение алгоритмов

Сравнение проводилось с использованием программы, написанной на языке С. В эксперименте были использованы 100 случайных не ориентированных графов, имеющих до 12 вершин. В табл. 1 представлены результаты экспериментов.

Таблица 1. Результаты экспериментов

n вершин	Среднее время перебора, сек	
	Полный	Неявный
10	0,177 (≤0,550)	0,001
11	1,04 (≤3,00)	0,001
12	7,6 (≤185,0)	0,002
13	103	0,003
14	321	0,003
15	372,0 (≤1649,5)	0,003
16	372	0,003

На рис. 4 представлены результаты экспериментов. График 1 иллюстрирует точность определения хроматического числа неявным перебором, а график 2 – зависимости времени перебора от количества входных данных. Если для наглядности в качестве времени работы использовался максимальный результат из зафиксированных при проведении измерения, то он добавлен в таблицу 1 (число в скобках).



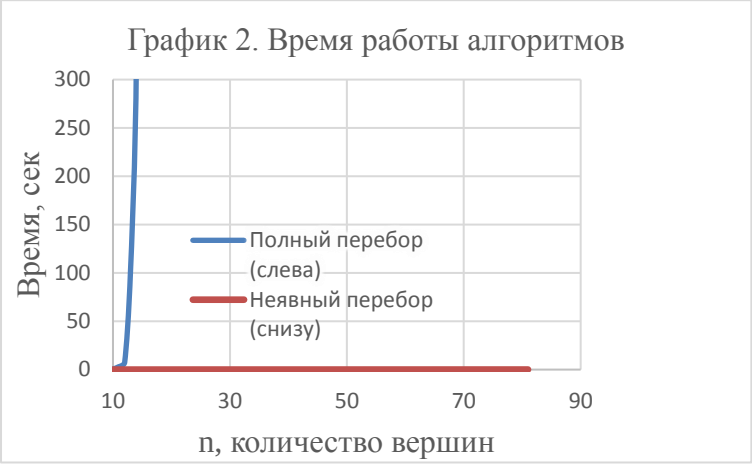


Рис. 4. Результаты экспериментов по раскраске графов различными алгоритмами

2. Применение алгоритмов раскраски графов.

2.1. Решение и составление судоку

Для составления судоку 4 на 4 необходимо 16 цифр. Пусть вершины графа - клетки таблицы, их 16. Тогда значением цвета вершины является цифра, которую необходимо записать для решения судоку.

На рис. 5 представлен граф решений и соответствующая ему таблица 2. Одинаковые цвета в таблице можно заменить одинаковыми цифрами от 1 до 4.

Решение для судоку размером 9 на 9 с помощью алгоритма неявного перебора найти не удалось. В лучшем случае машина выдавала ответ из 10 цветов.

Интересно отметить, что не стоит решать судоку методом перебора всех возможных решений, т.к. для поиска ответа алгоритмом полного перебора потребовалось бы совершить около 9^{81} перестановок.

Таблица 2. Таблица решений

К	*О*	*Ж*	*З*
Ж	*З*	*К*	*О*
О	*К*	*З*	*Ж*
З	*Ж*	*О*	*К*

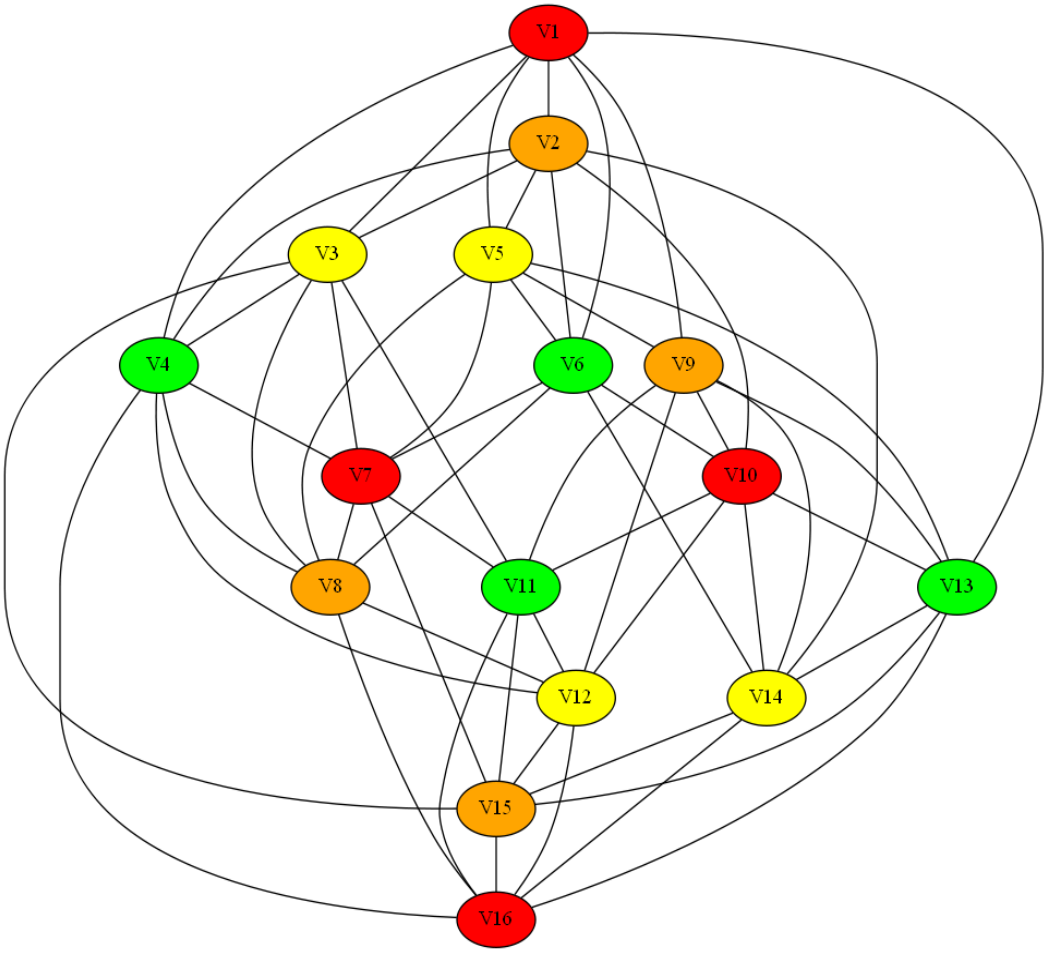


Рис. 5. Граф Судоку 4 на 4

2.2. Составление расписания

Пусть надо составить расписание 8 пар. Допустим, в одном потоке 2 группы, каждой группе необходимо посетить 5 занятий, из которых 3 семинара и 2 лекции, общие для обеих групп. Предположим, преподаватель М проводит семинары по математическому анализу, аналитической геометрии и читает одну лекцию, а преподаватель Г ведет занятия по начертательной геометрии и читает одну лекцию. Необходимо найти минимальное количество часов, за которое могут пройти все занятия.

Будем обозначать первой буквой предмет, который ведет один и тот же преподаватель, далее номер вершины по порядку и группа из потока, у которой ведут занятие. Пронумеруем вершины графа *G*:

- математический анализ: М1-1гр и М2-2гр;
- аналитическая геометрия: М3-1гр и М4-2гр;
- общая лекция по мат. анализу: М5-Общ;
- начертательная геометрия: Г6-1гр и Г7-2гр;
- общая лекция по нач. геометрии: Г8-общ.

Составим граф так, что если занятие у одного преподавателя или одной группы, то они не могут проходить в одно время, и связаны ребром. Вершины, где находится группа

1, связаны между собой. Аналогично связаны вершины с группой 2 и вершины, содержащие одного преподавателя.

Результат работы для обоих рассмотренных выше алгоритмов один и тот же: необходимо минимум 6 часов, чтобы провести все занятия (рис. 6). Вершины, соответствующие семинарам с разными группами и преподавателями, окрашены в один цвет: Г6 и М2 – 2 цвет, М1 и Г7 – 1 цвет.

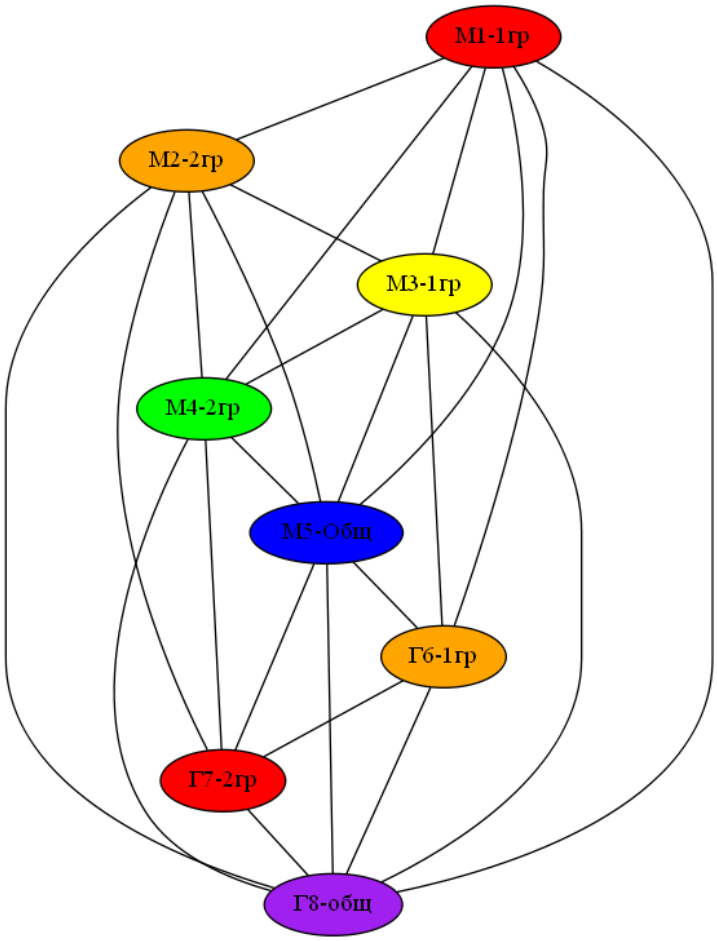


Рис. 6. Граф для составления расписания

2.3. Микроэлектроника

Структура микроэлектронных твердотельных устройствах наноразмерного масштаба состоит из большого количества материалов с разными свойствами, зачастую диаметрально противоположными. Границы этих материалов (например, изолятор двуокись кремния) примыкают к проводящим областям (например, легированный кремний), а с ними связан по границе проводящий материал (например, золото или алюминий) (рис. 7).

Структуры эти весьма сложные и создать обозримую модель для исключения контактов нежелательных материалов стоит большого труда [5]. Алгоритмы раскраски графов могут быть использованы для проработки и выявления нежелательных граничных контактов, приводящих к браку. На рис. 8 приведен пример раскраски графа полупроводникового транзистора.

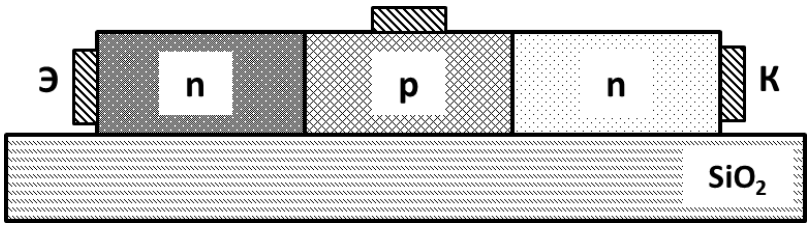


Рис. 7. Структура полупроводникового транзистора

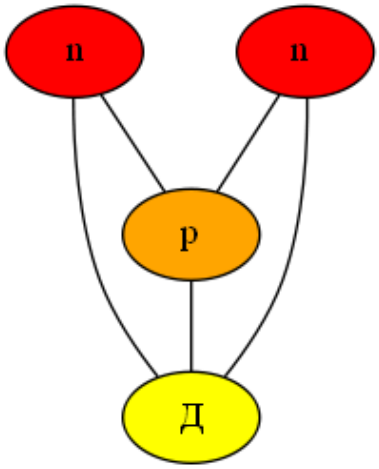


Рис. 8. Раскраска графа полупроводникового транзистора: цвет 1 - полупроводник *n*-типа (Э, К); цвет 2 - полупроводник *p*-типа (Б); цвет 3 - диэлектрик (SiO_2)

Заключение

По результатам сравнительного анализа алгоритмов неявного и полного перебора было установлено, что метод полного перебора ограничен применением на практике высокой вычислительной сложностью, в отличие от достаточно эффективного метода неявного перебора. Средняя точность определения минимально возможного цвета графа с помощью неявного перебора составляет 86 %. Однако в ходе эксперимента было замечено, что для больших графов точность снижается. Также зафиксированы частые отклонения от точного значения для графов высокой плотности (с высокой степенью вершин).

Самая большая зафиксированная разница с точным значением равна двум. Таким образом, определяемое количество цветов имеет значение не более, чем $\chi(G)+2$.

Эффективность работы алгоритма полного перебора имеет неплохие показатели для малых графов, но с увеличением количества входных данных временные затраты становятся слишком большими. Тогда как алгоритм неявного перебора помогает найти хорошее приближение хроматического числа за считанные секунды. Несмотря на то, что такой алгоритм не всегда достаточно точен, он часто находят применение благодаря легкой реализации и высокой скорости работы.

Необходимо отметить, что для некоторых других графов, не рассмотренных в статье, хроматическое число, найденное неявным методом, сильно отличается от их нижних оценок [2].

Таким образом, для практических применений, например, при синтезе ДНК и поиске бихроматических фрагментов [6] перспективным представляется использование алгоритма неявного перебора.

Работа выполнена при частичной финансовой поддержке гранта РФФИ 15-29-01115 офи-м.

Список литературы

[1]. Емеличев В.А., Мельников О.И., Сарванов В.И., Тышкевич Р.И. Лекции по теории графов. М: Наука, 1990. 384 с.

[2]. Кристофидес Н. Теория графов: Алгоритмический подход: пер. с англ. Э.В. Вершкова, И.В, Коновальцева / под ред. Г.П. Гаврилова. М: Мир, 1978. 432 с. [Nicos Christofides. Graph Theory: An Algorithmic Approach. New York: Academic Press, 1975].

[3]. Оре О. Теория графов: пер. с англ. И.Н. Врублевской / под ред. Н.Н. Воробьева. М: Наука, 1968. 352 с. [Oysten Ore. Theory of Graphs. American Mathematical Society, 1962].

[4]. Кормен Т.Х., Лейзерсон Ч.И., Ривест Р.Л., Штайн К. Алгоритмы: построение и анализ: пер. с англ. 2-е изд. М.: Вильямс, 2005. 1290 с. [Cormen, T.H., Leiserson C.E, Rivest R.L., Stein C. Introduction to Algorithms. 2nd. MIT Press and McGraw-Hill, 2001].

[5]. Сокол В.А. Электрохимическая технология микро- и наноэлектронных устройств // Доклады БГУИР. 2004. № 3. С. 18-26.

[6]. Abfalter I., Flamm C., Stadler P. Design of Multi-Stable Nucleic Acid Sequences // Proceedings of the German Conference on Bioinformatics (Neuherberg/Garching near Munich, Germany, October 12-14, 2003). Oxford University Press, 2004. Vol. 1. P. 1-7.