

**АЛГОРИТМЫ РЕШЕНИЯ ЗАДАЧ
ДИСКРЕТНОЙ МАТЕМАТИКИ**



Сибирский ордена Трудового Красного Знамени
физико-технический институт им. В.Д.
Кузнецова при Томском государственном
университете

АЛГОРИТМЫ РЕШЕНИЯ
ЗАДАЧ ДИСКРЕТНОЙ МАТЕМАТИКИ

Выпуск 2

Под редакцией Г.П.Агибалова

Издательство Томского университета

Томск - 1987

УДК 519.7.681.3

Алгоритмы решения задач дискретной математики, вып.2.-
Томск: Изд-во Томского ун-та, 1987. - 203 с. I р. 60 к. 500 экз.
1502000000.

Представлены результаты исследований по дискретной математике и ее приложениям в системах автоматизированного проектирования цифровых автоматов на базе интегральных схем, в том числе элементы теории функциональных систем на полурешетках, условия существования некоторых нетривиальных декомпозиций конечных автоматов, алгоритмы синтеза и имитационного моделирования функциональных схем и микропроцессорных систем, алгоритмы обхода и раскраски графов, алгоритмы конструирования схем, включая их декомпозицию, размещение и трассировку, алгоритмы технической диагностики и некоторые другие.

Для специалистов по дискретной математике и автоматизированному проектированию цифровой электронной техники.

Рецензент - С.В.Быкова

А 1502000000 40-87
177(012)-87

© Издательство Томского университета, 1987

В.А.Беляев

АЛГОРИТМ РАСКРАСКИ ГРАФА МЕТОДОМ СОКРАЩЁННОГО ОБХОДА ДЕРЕВА ПОИСКА

Задача раскраски графа находит многочисленные приложения на различных этапах проектирования цифровой аппаратуры. В конструкторском проектировании задача раскраски возникает в связи с вопросами компоновки, распределения по слоям и трассировки межсоединений элементов схем. В логическом проектировании она возникает при минимизации числа состояний автомата, при кодировании его внутренних состояний и при получении минимальных д.н.ф. булевых функций [1]. В теоретическом программировании к ней сводится задача получения наиболее экономного распределения памяти в операторных схемах программ [2, 3]. Известная трудоёмкость существующих алгоритмов решения этих задач практической сложности побуждает к изысканию новых подходов и методов, совершенствованию известных алгоритмов. В работе предлагается усовершенствованный алгоритм решения задачи раскраски графа методом сокращённого обхода дерева поиска [4]. Усовершенствование проводится по двум путям: во-первых, в случае существования в заданном графе разделяющего полного подграфа решение задачи раскраски всего графа заменяется решением более простых задач раскраски для покрывающих подграфов с меньшим числом вершин, во-вторых, вводятся более эффективные, чем в [4], параметры метода сокращённого обхода.

Задача раскраски графа ставится следующим образом: для заданного симметрического графа $G=(V, U)$ требуется указать такое разбиение множества вершин V , называемое минимальной раскраской графа G , в котором число классов (красок) минимально и каждый класс есть внутренне устойчивое множество (ВУМ), т.е. множество с попарно несмежными вершинами.

Введём необходимые определения и понятия. Говорят, что вершина $x \in V$ связного графа $G=(V, U)$ является точкой сочленения, если подграф, получаемый удалением x , несвязен [5]. Полный подграф H в графе G называется разделяющим полным подграфом, если подграф, получаемый удалением H из G , несвязен. По определению, одновершинный разделяющий подграф является точкой сочленения. Пусть H — разделяющий полный подграф в графе G и $G'_1, \dots, G'_k$, $k \geq 2$ подграфы, являющиеся компонентами связности в $G \setminus H$. Обозначим через $R_1, \dots, R_k$ минимальные раскраски подграфов

$G_1 = G'_1 \cup H, \dots, G_k = G'_k \cup H$ соответственно, а через $\gamma(G)$ — хроматическое число графа G , т.е. число классов (красок) в минимальной раскраске графа G , тогда справедливы следующие соотношения:

$$\gamma(G) \geq \max(\gamma(G_1), \dots, \gamma(G_k)), \quad (I)$$

$$\gamma(G_1), \dots, \gamma(G_k) \geq \gamma(H).$$

Из определения раскраски следует, что все вершины полного подграфа H содержатся в разных блоках разбиений $R_1, \dots, R_k$. Пусть $\gamma(H) = h$; перенумеруем все вершины в H натуральными числами $1, 2, \dots, h$ и в соответствии с этой нумерацией упорядочим блоки в разбиениях $R_1, \dots, R_k$ так, чтобы блок, содержащий вершину i из H , имел номер $i \in \{1, \dots, h\}$ в своем разбиении, порядок остальных блоков (если они имеются) в соответствующих разбиениях $R_1, \dots, R_k$ произволен. Упорядоченные таким образом разбиения обозначим через $R_1^*, \dots, R_k^*$ соответственно. Пусть $t = \max(|R_1|, \dots, |R_k|)$ и пусть для определенности $R_i^* = (A_1^i, \dots, A_{t_i}^i)$, $i \in \{1, \dots, k\}$, тогда построим множество $R^* = \{B_1, \dots, B_t\}$, в котором каждый элемент B_j , $j \in \{1, \dots, t\}$ есть объединение классов с номерами j из разбиений $R_1^*, \dots, R_k^*$, т.е.

$$B_j = \bigcup_{i=1}^k A_j^i, \quad j \in \{1, \dots, t\} \text{ и } A_j^i = \emptyset \text{ для } j > t_i \in R_i. \quad (2)$$

Непосредственно из построения следует, что множество $R^* = \{B_1, \dots, B_t\}$ образует разбиение множества V вершин графа G . В дальнейшем процесс получения разбиения R^* по правилу (2) будем называть склеиванием разбиений $R_1, \dots, R_k$, а само разбиение R^* — результатом склеивания.

Имеет место следующая теорема.

Теорема I. Множество $R^* = \{B_1, \dots, B_t\}$ есть минимальная раскраска графа G .

Сначала покажем, что каждый блок B_j , $j \in \{1, \dots, t\}$, разбиения R^* есть внутренне устойчивое множество в графе G . Из определения разделяющего полного подграфа $H \subset G$ следует, что объединение любых внутренних устойчивых множеств, взятых по одному из разных компонент связности графа $G \setminus H$, является внутренне устойчивым множеством в графе G . В дальнейшем будем называть такое объединение ВУМ типа I. Рассмотрим набор B блоков, взятых по одному из каждого разбиения-раскраски $R_1, \dots, R_k$ соответствующих графов $G'_1 \cup H, \dots, G'_k \cup H$ и содержащих одну и ту же вершину a полного графа H . Удалив вершину a из каждого класса набора B , получим

набор внутренне устойчивых множеств из разных компонент связности $G'_1, \dots, G'_k$, объединение которых есть ВУМ типа 1. В силу того, что вершина a графа H совместима с каждым подмножеством в наборе B , она совместима и с объединением этих подмножеств, т.е. объединение классов в наборе B есть внутренне устойчивое множество, назовём его ВУМ типа 2. Из правила (2) построения блоков разбиения R^* следует, что каждый блок B_j является ВУМ типа 1 или ВУМ типа 2, т.е. состоит из попарно несмежных вершин в G . Следовательно, каждый класс разбиения R^* есть внутренне устойчивое множество и R^* есть раскраска графа G .

По построению число блоков в R^* равно t , следовательно, на основании (1) R^* есть минимальная раскраска графа G .

Таким образом, процесс минимальной раскраски заданного графа G в случае существования в нём разделяющего полного подграфа H предлагается разлагать на следующие этапы:

- нахождение компонент связности $G'_1, \dots, G'_k$ графа $G \setminus H$;
- нахождение минимальных раскрасок $R_1, \dots, R_k$ для каждого из графов $G'_1 \cup H, \dots, G'_k \cup H$;
- получение минимальной раскраски R^* графа G путём склеивания разбиений-раскрасок $R_1, \dots, R_k$ по правилу (2).

Преимущество такого подхода подтверждают следующие рассуждения. Поскольку объём перебора $\delta(n, m)$ при решении задачи раскраски с ростом числа n и m соответственно вершин и рёбер в графе G растёт нелинейно, то суммарный объём вычислений, затрачиваемый для раскраски подграфов $G_1, \dots, G_k$, покрывающих G , значительно ниже, чем $\delta(n, m)$. Следовательно, располагая эффективным алгоритмом нахождения разделяющего полного подграфа в заданном графе G , удастся значительно сократить объём вычислений, заменяя задачу раскраски графа G задачами раскраски графов $G_1, \dots, G_k$ с меньшим числом вершин, которые могут быть решены любым известным алгоритмом.

В [4] задача раскраски графа решается методом сокращённого обхода дерева поиска, в качестве параметров используется алгоритм Ψ построения максимальных внутренне устойчивых множеств, операция Ψ сокращения и функция F_0 нижней оценки, которые в основном определяют эффективность метода в целом. В данной работе предлагаются улучшенные параметры, повышающие эффективность метода сокращённого обхода для задачи раскраски графа.

Ниже описывается более эффективный, чем в [4], алгоритм Ψ_1 построения множества всех максимальных внутренне устойчивых под-

множеств вершин заданного графа $G = (V, E)$, содержащих некоторую фиксированную вершину v из V ; при изложении алгоритма предполагается, что вершины графа G упорядочены некоторым образом и образуют ряд $v_1, \dots, v_n$, в котором $v_i = v$. Пусть также $O_i(A)$ обозначает множество всех вершин в G , смежных вершинам из A . Описываемый алгоритм осуществляет направленный перебор по дереву и является модификацией алгоритма построения максимальных независимых множеств из [6]. Работа алгоритма состоит в последовательном выполнении шагов ветвления и возвращения, причём шаг возвращения выполняется лишь тогда, когда невозможен шаг ветвления. В процессе поиска на некотором шаге i рассматриваются три множества: A_i - строящееся внутренне устойчивое множество; B_i - наибольшее множество совместимых с A_i вершин такое, что $A_i \cap B_i = \emptyset$, и после добавления любой вершины из B_i к A_i получается новое совместимое множество A_{i+1} , т.е. B_i состоит из вершин графа G , являющихся кандидатами на расширение множества A_i и, наконец, множество C_i содержит все несовместимые с A_i вершины из G , т.е. $C_i = O_i(A)$. Тогда шаг ветвления в дереве поиска состоит в выборе вершины $v_{i+1} \in B_i$, включении её в A_i для построения множества $A_{i+1} = A_i \cup \{v_{i+1}\}$, вычислении новых множеств: $B_{i+1} = B_i - \{v_{i+1}\} \cup O(v_{i+1})$, $C_{i+1} = C_i \cup O(v_{i+1})$. Шаг возвращения алгоритма η состоит в возвращении к исходному множеству A_i и удалении вершин v_{i+1} из B_i . Нетрудно видеть, что множество A_i на некотором шаге i является максимальным тогда и только тогда, когда

$$B_i = \emptyset \quad \text{и} \quad A_i \cup C_i = V. \quad (3)$$

В [6] предложено достаточное условие для осуществления шага возвращения, заключающееся в следующем. Пусть B_i^* подмножество тех вершин из первоначального B_i , которые уже использовались для расширения A_i и $B_i^+ -$ те вершины из B_i , которые ещё не использовались. Тогда условие существования в B_i^* вершины v такой, что

$$O_i(v) \cap B_i^+ = \emptyset, \quad (4)$$

является достаточным для осуществления шага возвращения. Данное условие в алгоритме η удобнее использовать при выполнении шага ветвления в следующем виде. Пусть $v_{i+1} \in B_i$ вершина, выбранная для ветвления и v_i вершина с максимальным номером в множестве $O_i(v_{i+1})$, тогда, удалив из B_i вершины с номерами $i+1, \dots, n$, мы тем самым исключим из кандидатов на расширение множества A_i все те вершины v , для которых выполнено условие (4), поскольку любое расширение множества $A_{i+1} = A_i \cup \{v\}$ не является максимальным внутренне устойчивым.

состоит из следующих подмножеств в V : $\{1, 3, 5, 9\}$, $\{1, 3, 5, 10\}$, $\{1, 3, 8, 9\}$, $\{1, 6, 8, 9\}$. Для простоты изображения на рис.2 множество $\{a, b, c\}$ обозначается без скобок через a, b, c .

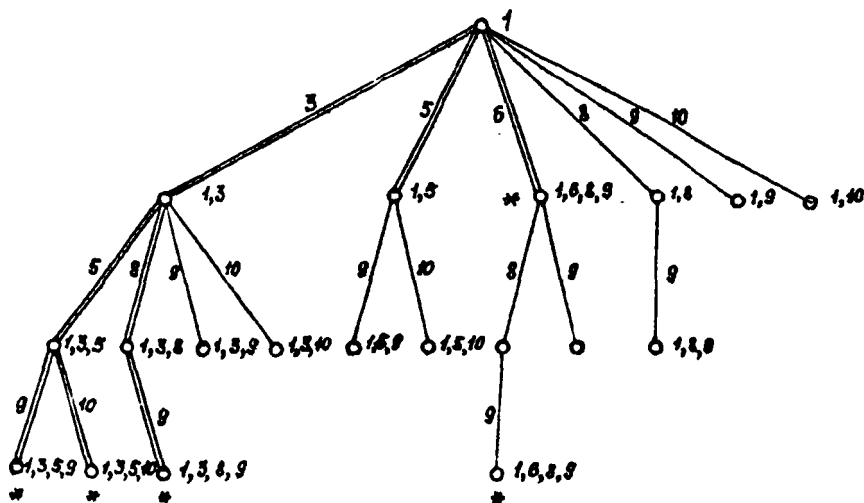


Рис.2

За функцию нижней оценки F_0 принимается число $h = \max(a, b)$, где a есть наибольшее число вершин в максимальном полном подграфе в G (кликовое число G [6]) и $b = \lfloor \frac{n}{2} \rfloor$, n - число вершин в графе G , а c - число независимости G [6] (или кликовое число дополнительного графа $\bar{G}$).

Кликовое число χ для любого графа находится следующим быстродействующим алгоритмом χ_0 . При изложении алгоритма предполагается, как и раньше, что вершины в графе G упорядочены и образуют ряд $v_1, \dots, v_n$ и граф G не пустой ($h \geq 2$), т.е. содержит по крайней мере одно ребро. Алгоритм осуществляет направленный перебор с использованием дерева поиска. В алгоритме осуществляется последовательное выполнение шагов двух типов - ветвления и возвращения. Причем шаг ветвления выполняется всегда, когда невозможен шаг возврата. На каждом шаге δ ветвления, рассматривается текущее множество вершин D_δ ($D_1 = V, \delta = 1$), в котором выбирается вершина с наименьшим номером, строится множество $D_{\delta+1} = D_\delta \cup \{u\}$ и, если $|D_{\delta+1}| + \delta \leq \chi$, выполняется шаг возвращения, заключающийся в вычислении новых $D_\delta = D_\delta - \{u\}$ и $\delta = \delta - 1$ и переходе к выполнению шага δ .

Кроме того, при выполнении очередного шага ветвления сократить число ветвлений можно включая в A_3 все те вершины v из B_3 , для которых справедливо

$$O_i(v) \subseteq C_3, \quad (5)$$

поскольку отсутствие любой такой вершины в расширении множества приведёт к нарушению условия (3) максимальности.

Ниже даётся формальное описание алгоритма.

Алгоритм $\mathcal{A}$.

1. $A_1 := \{v_1\}$, $C_1 := A_1 \cup O_1(A_1)$, $B_1 := V - C_1$, $D_1 := B_1$, $\delta := 0$.
2. Положим $\delta := \delta + 1$ и, если $B_\delta = \emptyset$, выполним п.4, иначе п.3.
3. $D_\delta := B_\delta$ и, если не существует вершины $v \in B_\delta$ со свойством $O_i(v) \subseteq C_\delta$, выполним п.5, иначе положим $A_{\delta+1} := A_\delta \cup \{v\}$, $B_{\delta+1} := B_\delta - \{v\}$ и, в случае $B_\delta = \emptyset$, выполним п.3, а при $B_\delta \neq \emptyset$ - п.4.
4. Если $A_{\delta+1} \cup C_{\delta+1} \neq V$, то п.5, иначе $A_{\delta+1}$ отмечается как очередной элемент перечисляемого множества и п.5.
5. Если $D_\delta = \emptyset$, положим $\delta := \delta - 1$ и п.6, иначе пусть u есть вершина с наименьшим номером в D_δ , а v_2 - вершина с наибольшим номером в $O_i(u)$, тогда положим $A_{\delta+1} := A_\delta \cup \{u\}$, $B_{\delta+1} := B_\delta - \{u\}$, $C_{\delta+1} := C_\delta \cup \{u\} \cup O_i(u)$, $D_{\delta+1} := D_\delta - \{u, v_2, \dots, v_k\}$, $B_{\delta+1} := B_\delta - C_{\delta+1}$ и выполним п.2.
6. Если $\delta > 0$, выполним п.5, в противном случае алгоритм своей работы заканчивает.

Пример 1. Проиллюстрируем работу описанного алгоритма $\mathcal{A}$. Построим множество всех максимальных внутренне устойчивых множеств (МВУМ)

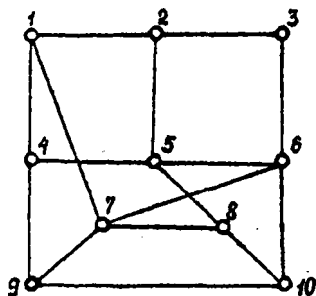


Рис. 1

Корень дерева на рис. 2 соответствует множеству $A_1 = \{1\}$, а концевые вершины, отмеченные знаком (x), представляют перечисляемые МВУМ. Каждое ребро помечено вершиной u , выбираемой для расширения A_3 при реализации соответствующего шага ветвления в алгоритме $\mathcal{A}$, а каждая вершина помечена соответствующим множеством A_3 . Таким образом, множество всех максимальных совместимых подмножеств

для графа G из [7] на рис. 1 с множеством вершин $V = \{1, \dots, 10\}$ и начальной фиксированной вершиной $v = 1$. На рис. 2 изображено дерево перебора D , которое обходит алгоритм $\mathcal{A}$ из [4] в процессе построения МВУМ для данного примера; двойными линиями изображена та часть дерева D , которую обходит описанный выше алгоритм $\mathcal{A}$ при построении того же множества.

ветвления; в противном случае выполняется очередной шаг ветвления. В случае $|D_{\delta+1}| = \emptyset$ величина γ принимается равной δ и выполняется шаг возврата. После завершения работы алгоритма величина γ есть кликовое число графа G .

Алгоритм γ_0 .

1. $\delta := 1, \gamma := 2, D_1 := \{v_1, \dots, v_n\}$.
2. Если $D_\delta = \emptyset$, то п.3, иначе в множестве D_δ выбрать вершину v_δ с наименьшим номером, положить $D_\delta := D_\delta - \{v_\delta\}, D_{\delta+1} := D_\delta \cap Q(v_\delta)$ и, если $|D_{\delta+1}| + \delta \leq \gamma$, то п.2, в противном случае положить $\delta := \delta + 1$ и, если $D_\delta \neq \emptyset$, выполнить п.2, иначе положить $\gamma := \delta - 1$ и п.3.
3. Положить $\delta := \delta - 1$ и при $\delta \neq 0$, если $|D_\delta| + \delta > \gamma$, выполним п.2, а при $|D_\delta| + \delta \leq \gamma$ - п.3; при $\delta = 0$ алгоритм работу заканчивает, значение γ есть кликовое число графа G .

Для нахождения C -числа независимости графа G достаточно применить алгоритм γ_0 к графу $\bar{G}$.

Например, применив описанный алгоритм к графу G из [6] на рис.3 с множеством вершин $V = \{1, \dots, 16\}$ получим кликовое число $\alpha = 2$, применив алгоритм к графу $\bar{G}$ получим кликовое число дополненного графа $\alpha = 5$; в результате имеем $P_0 = \max(2, \lceil 16/5 \rceil) = 4$.

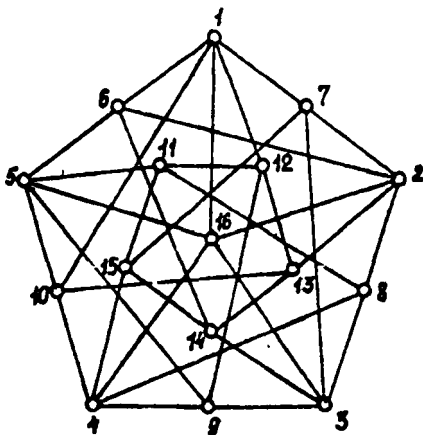


Рис. 3

Применение указанных параметров позволяет находить минимальную раскраску графов методом сокращённого обхода дерева поиска с меньшими вычислительными затратами.

Л и т е р а т у р а

1. Фрицнович Г.Ф. Синтез асинхронных конечных автоматов и раскраска вершин графов. - В кн.: II Всесоюзное совещание по теории релейных устройств и конечных автоматов. Тезисы докладов. Рига, 28 сент. - I окт. 1971, с. 33 - 34.
2. Бухгольц Н.В., Дьяченко В.Ф., Лазарев В.Г. и др. Алгоритмы минимизации оперативной памяти ЦВМ. - В кн.: Проблемы синтеза цифровых автоматов. М.: Наука, 1967, с. 112 - 118.
3. Вршов А.П., Кожухин Г.И. Об оценках хроматического числа связанных графов. - ДАН СССР, 1962, т.142, №2, с. 270 - 273.
4. Агибалов Г.П., Беляев В.А. Технология решения комбинаторно-логических задач методом сокращённого обхода дерева поиска. Томск, Изд-во Том. ун-та, 1981. - 125 с.
5. Берж К. Теория графов и её применения. - М.: ИЛ, 1962. - 320 с.
6. Кристофидес Н. Теория графов. Алгоритмический подход. - М.: Мир, 1976. - 432 с.
7. Wilkov R.S., Kim W.H. A Practical Approach to the Chromatic Partition Problem. - J. Franklin Inst., 1970, 289, №5, p. 333 - 349.