OSTIS-2013
(Open Semantic Technologies for Intelligent Systems)

УДК 519.7

ЗАДАЧИ НА ГРАФАХ В ТЕОРИИ ПРОЕКТИРОВАНИЯ ЦИФРОВЫХ УСТРОЙСТВ

Поттосин Ю.В.[*], Поттосина С.А.[**]

[*] *Объединенный институт проблем информатики НАН Беларуси, г. Минск, Республика Беларусь*

pott@newman.bas-net.by

[**] *Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь*

pottosina@sum-solution.net

Рассматриваются задачи на графах, имеющие приложение в проектировании дискретных устройств управления. Дается обзор методов и алгоритмов решения этих задач. Приводятся примеры использования графовых моделей для решения задач логического проектирования.
Ключевые слова: логическое проектирование; раскраска графа; полный подграф; двудольный подграф.

Библиотека БГУИР

Введение

Объектно-ориентированный подход к решению проблем логического проектирования дискретных устройств предполагает построение базы знаний. База знаний содержит структурированную информацию и правила вывода, которые покрывают библиотечными элементами данную область знаний. При логическом проектировании дискретных устройств такими элементами могут служить логико-комбинаторные задачи и алгоритмы их решений, сформулированные в терминах теории графов.

Методы теории графов широко применяются в логическом проектировании дискретных управляющих и вычислительных устройств. О приложениях теории графов в логическом проектировании и близких к нему областях написано довольно много работ, перечислить которые было бы сложно, поэтому ограничимся только указанием некоторых монографий [Мелихов, 1971], [Мелихов и др., 1974], [Нечепуренко и др., 1990] и обзорных статей [Закревский и др., 1986], [Поттосин, 2001].

На этапе логического проектирования дискретных устройств управления возникают разнообразные логико-комбинаторные задачи [Агибалов, 1981], [Закревский и др., 2007], [Поттосин, 2011], решение которых приводит к оптимальным вариантам логического проекта. В настоящей работе предлагается краткий обзор тех задач и методов теории графов, которые находят широкое применение в логическом проектировании.

В рассматриваемых задачах понятие графа используется в разных интерпретациях. Чаще всего это графическое представление бинарного отношения типа совместимости, т.е. отношения, обладающего свойствами рефлексивности и симметричности. Такое отношение представляется произвольным неориентированным графом. В некоторых случаях граф представляет отношение квазипорядка или отношение следования между некоторыми событиями. В отдельных задачах граф представляет логическую сеть. Рассматриваемые в этих случаях графы являются ориентированными.

1. Раскраска графа

Значительное место в логическом проектировании занимает задача раскраски графа. Раскраска графа может быть использована при решении задачи декомпозиции булевых функций [Perkowski, 1995], [Закревский, 2000], [Бибило, 1987], занимающей важное место в синтезе комбинационных схем в базисе сверхбольших интегральных схем (СБИС). Из всех задач логического проектирования, пожалуй, именно она наиболее часто сводится к раскраске графа.

Одна из постановок задачи декомпозиции формулируется следующим образом. Задана система булевых функций как векторная булева

функция $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, и заданы векторные булевы переменные $\boldsymbol{w}$ и $\boldsymbol{z}$, составленные из компонент векторной переменной $\boldsymbol{x}$. Необходимо представить заданную систему в виде $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{w}, \boldsymbol{g}(\boldsymbol{z}))$. Функции $\boldsymbol{h}$ и $\boldsymbol{g}$ должны быть более простыми, чем $\boldsymbol{f}$. В данном случае это означает, что число s компонент вектора $\boldsymbol{z}$ и сумма чисел компонент векторов $\boldsymbol{w}$ и $\boldsymbol{u} = \boldsymbol{g}(\boldsymbol{z})$ меньше n. Обычно стремятся получить минимум этой суммы.

Общий метод решения данной задачи предполагает построение графа G, вершинами которого являются значения $\boldsymbol{z}^*$ векторной переменной $\boldsymbol{z}$, а ребра определяются следующим образом. Пусть $\boldsymbol{x}^*_i$ – значение векторной переменной $\boldsymbol{x}$, а $\boldsymbol{z}^*_i$ и $\boldsymbol{w}^*$ – значения векторных переменных $\boldsymbol{z}$ и $\boldsymbol{w}$ соответственно, причем компоненты $\boldsymbol{x}^*$ совпадают с соответствующими компонентами $\boldsymbol{z}^*_i$ и $\boldsymbol{w}^*$. Вершины $\boldsymbol{z}^*_i$ и $\boldsymbol{z}^*_j$ графа G связаны ребром, если найдется значение $\boldsymbol{w}^*$ вектора $\boldsymbol{w}$ такое, что значения функции $\boldsymbol{f}$ при соответствующих $\boldsymbol{x}^*_i$ и $\boldsymbol{x}^*_j$ различны.

Легко доказывается, что заданную векторную функцию можно представить в виде $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{w}, \boldsymbol{u})$, где векторный аргумент $\boldsymbol{u}$ имеет k компонент, тогда и только тогда, когда хроматическое число графа G не превосходит 2^k.

Для получения окончательного решения достаточно получить минимальную раскраску графа G и закодировать полученные цвета булевыми векторами, которые представят значения функции $\boldsymbol{g}$ при соответствующих значениях вектора $\boldsymbol{z}$.

Ввиду того, что задача раскраски графа является NP-трудной и точное ее решение не всегда практически достижимо, в работе [Закревский и др., 2007] предлагается эвристический метод, который позволяет получать раскраску, довольно близкую к минимальной за практически приемлемое время. Алгоритм, реализующий данный метод, позволяет в отдельных случаях по характеру его выполнения показывать, что полученное решение является точным.

2. Кодирование графа. Двудольные подграфы

Как было сказано выше, при декомпозиции булевых функций в конечном счете требуется закодировать булевыми или троичными векторами цвета, полученные после раскраски графа. Данные векторы считаются значениями векторной переменной $\boldsymbol{u}$, обладающей тем свойством, что каждому ребру графа должна соответствовать ее компонента, принимающая противоположные значения (0 и 1) на концах этого ребра. Таким образом, каждой компоненте вектора $\boldsymbol{u}$ можно поставить в соответствие двудольный подграф рассматриваемого графа, а данную задачу можно рассматривать как задачу покрытия графа двудольными подграфами. Такой подход к решению задачи декомпозиции булевых функций описан в работах [Закревский, 2000], [Бибило, 1987].

Задача покрытия графа двудольными подграфами возникает при поиске оптимального кодирования состояний синхронного автомата, описанном в работе [Закревский и др., 2007]. Предполагается задание входа и выхода автомата в структурном алфавите. Такая модель названа в работе [Закревский, 1981] автоматом с абстрактным состоянием. Автомат представляется в виде двух троичных матриц $\boldsymbol{U}$ и $\boldsymbol{V}$. Столбцам матрицы $\boldsymbol{U}$ соответствуют входные булевы переменные $x_1, x_2, \ldots, x_n$, столбцам матрицы $\boldsymbol{V}$ – выходные булевы переменные $y_1, y_2, \ldots, y_m$. Строки обеих матриц помечены состояниями автомата так, что пара их одноименных строк представляет переход из состояния, являющегося меткой строки матрицы $\boldsymbol{U}$, в другое состояние, являющееся меткой строки матрицы $\boldsymbol{V}$. Строка матрицы $\boldsymbol{U}$ представляет при этом условие перехода, а строка матрицы $\boldsymbol{V}$ – двоичные выходные сигналы, сопровождающие этот переход.

Метод заключается в следующем. Для некоторого столбца y_i матрицы $\boldsymbol{V}$ строится граф G, вершинам которого соответствуют состояния заданного автомата. В данном столбце y_i отыскиваются пары элементов с противоположными значениями (0 и 1), которым соответствуют пары неортогональных строк матрицы $\boldsymbol{U}$. Вершины графа G, соответствующие меткам строк всякой такой пары, связываются ребром и затем кодируются, как показано выше. Полученные коды переносятся в матрицы $\boldsymbol{U}$ и $\boldsymbol{V}$ в виде значений вновь введенных внутренних переменных $z_1, z_2, \ldots, z_k$. Эта процедура повторяется для каждого столбца матрицы $\boldsymbol{V}$, причем каждый раз выбирается столбец с минимальным числом пар указанных элементов. Когда граф G для каждого столбца оказывается пустым (после введения новых переменных строки становятся ортогональными), процесс заканчивается, а преобразованные матрицы $\boldsymbol{U}$ и $\boldsymbol{V}$ представляют систему булевых функций, описывающую комбинационную часть логической сети, реализующей заданный автомат. Некоторые совместимые состояния оказываются закодированными неортогональными или даже одинаковыми кодами. В этом случае фактическое число состояний автомата оказывается уменьшенным.

В работе [Поттосин, 1985] описан метод выделения в заданном графе G основного двудольного подграфа (т. е. такого подграфа, множество вершин которого совпадает с множеством вершин заданного графа) с множествами вершин V^0 и V^1 и максимальным числом ребер. Алгоритм, основанный на данном методе, состоит из двух этапов. На первом этапе находятся все циклы нечетной длины, на втором этапе решается задача покрытия множества полученных циклов ребрами графа. Ребра, составляющие найденное покрытие, затем удаляются, в результате чего получается искомый подграф. В зависимости от конкретной реализации

Библиотека БГУИР

данного метода решение может быть как точным, так и приближенным.

Эвристический метод решения этой задачи предложен в работе [Бибило, 1987] и уточнен в статье [Закревский, 2000]. Согласно этому методу начальное значение V^0 представляет одноэлементное множество, а V^1 содержит все остальные вершины. Затем вершины из множества V^1 последовательно переносятся в множество V^0 до тех пор, пока не перестанет увеличиваться число ребер, связывающих вершины из V^0 с вершинами из V^1.

3. Независимые множества. Полные подграфы

Задачи логического проектирования, формулируемые как задачи разбиения заданного множества на совместимые в некотором смысле подмножества, удается свести к раскраске графа, если попарной совместимости элементов подмножества достаточно для их групповой совместимости. В противном случае приходится вводить какие-то дополнительные условия одноцветности вершин. Другой подход к решению таких задач предполагает нахождение в заданном графе наибольшего независимого множества, т.е. множества попарно несмежных вершин, а затем группирование около них остальных вершин заданного графа.

С помощью этого подхода можно решать такие задачи, как минимизация длины кода состояний асинхронного автомата, сжатие таблицы переходов автомата (по строкам и столбцам), параллельная декомпозиция автомата, упрощение системы булевых функций, описывающих асинхронный автомат. Этот прием особенно подходит для решения таких задач абстрактного синтеза автомата, как минимизация числа состояний синхронного автомата и его декомпозиция. В этих задачах искомая совокупность совместимых подмножеств должна обладать свойством замкнутости, которое заключается в том, что наличие каких-либо элементов в одном совместимом подмножестве требует включения некоторых других элементов в другое совместимое подмножество из получаемой совокупности. Двойственной задачей по отношению к задаче нахождения независимых множеств является задача нахождения полных подграфов, поскольку полный подграф некоторого графа порождается независимым множеством его дополнения.

Для получения всех максимальных независимых множеств можно с успехом применять классический метод Полла-Ангера построения всех максимальных совместимых множеств состояний автомата. Алгоритм, основанный на этом методе, описан в терминах теории графов в работе [Закревский и др., 2007]. В нем используется прием, при котором заданный граф разлагается на последовательность подграфов и решение данной задачи для предыдущего подграфа преобразуется в решение для последующего подграфа. В этих же работах описан алгоритм получения всех максимальных независимых множеств, использующий лексикографический перебор.

4. Кодирование состояний автомата. Полный булев граф

В теории булевых функций используется представление булева пространства в виде графа, в котором вершинами являются всевозможные булевы вектора заданной размерности (элементы булева пространства) и две вершины связаны ребром тогда и только тогда, когда соответствующие векторы отличаются друг от друга значением только одной компоненты. В теории графов такой граф называется полным булевым графом и обозначается символом Q_k, где k – размерность соответствующего булева пространства.

Данная модель используется для решения задачи кодирования внутренних состояний дискретного автомата. Это необходимо для перехода от поведенческой модели автомата к его структурной модели. То есть каждое состояние автомата, представленное абстрактным символом в поведенческой модели, должно быть заменено булевым вектором, который, в свою очередь, представляет набор состояний двоичных элементов памяти в логической схеме, реализующей данный автомат. Критерием оптимизации при кодировании состояний автомата служит площадь кристалла, на котором размещается схема, или оценка величины энергии, которую может потреблять проектируемое устройство.

В работе [Закревский, 2005] процесс кодирования состояний автомата представляется как размещение состояний в булевом пространстве. Этот же процесс описан в работе [Закревский и др., 2007] как построение полного булева графа Q_k, напоминающее сборку некоторой простой механической конструкции.

Пусть задан автомат, множество состояний которого $Q = \{q_1, q_2, \dots, q_\gamma\}$. Вершины графа Q_k, являющиеся первоначально вершинами некоторого пустого графа (без ребер), заранее поставлены в соответствие состояниям автомата. На парах этих вершин $\langle q_i, q_j \rangle$ задана функция целочисленная w_{ij}, которая определяется в зависимости от применяемого критерия оптимизации. Если надо получить минимум площади кристалла, то функция w_{ij} определяется как в работе [Armstrong, 1962], и состояния в графе Q_k должны располагаться тем ближе, чем больше величина w_{ij}. Если надо получить минимум потребляемой энергии, то значение функции w_{ij} должно быть связано с частотой переходов между состояниями q_i и q_j.

Построение графа Q_k, который в теории булевых функций называется k-мерным гиперкубом, представляется как последовательность k шагов. На

Библиотека БГУИР

p-м шаге рассматривается множество $(p-1)$-мерных гиперкубов (графов Q_{p-1}), они объединяются в пары, и из каждой пары получается один p-мерный гиперкуб (граф Q_p) путем соответствующего добавления ребер. При этом по возможности для соединения ребрами выбираются те пары вершин, которым соответствуют наибольшие значения величины w_{ij}. Вершинам полученного графа Q_k приписываются k-компонентные булевы векторы с соблюдением отношения соседства, представленного ребрами графа Q_k.

5. Задачи на графах в синтезе структур СБИС

Довольно значительное многообразие задач на графах встречается при оптимизации структур СБИС. Задачи оптимизации структур СБИС лежат на стыке этапов логического и технического проектирования. При оптимизации структуры программируемой логической матрицы (ПЛМ) – одной из структур СБИС – особое место занимает задача свертки ПЛМ, которая заключается в использовании одной и той же линии (горизонтальной или вертикальной) для укладки различных проводников, что приводит к сокращению площади кристалла, занимаемой проектируемой схемой.

К раскраске графа в работе [Бибило, 1992] сводится получение многократной свертки ПЛМ, при которой на одной линии укладывается более чем два проводника, а внешние соединения переносятся в "третье измерение". Вершины рассматриваемого графа соответствуют проводникам, и две вершина связаны ребром, если и только если соответствующие проводники не могут быть уложены на одной линии. Раскраска графа указывает способ укладки проводников.

При заданном разбиении множества внешних сигналов на два подмножества, соответствующие сигналам, подводимым к ПЛМ с разных сторон, задача нахождения максимального числа свертываемых пар сводится к задаче нахождения максимального паросочетания в двудольном графе [Бибило, 1992]. Доли этого графа соответствуют упомянутым подмножествам, и две вершины из разных долей связаны ребром, если соответствующие сигналы образуют пару свертки.

При свертывании горизонтальных линий ПЛМ также решается задача размыкания контуров в двудольном орграфе, после чего надо должным образом упорядочить вертикальные шины. Это упорядочение сводится к задаче топологической сортировки вершин бесконтурного орграфа [Бибило, 1992]. Решением этой задачи является последовательность вершин, в которой для каждой дуги вершина, являющаяся концом этой дуги, расположена после вершины, являющейся ее началом.

В последнее время проектировщики электронной техники и исследователи в области проектирования дискретных устройств стали уделять много внимания сокращению расхода энергии при эксплуатации проектируемого устройства. Это обусловлено стремлением увеличить время действия источника энергии в портативных приборах, а также снизить остроту проблемы отвода тепла при проектировании сверхбольших интегральных схем. Поэтому одним из основных критериев оптимизации при проектировании дискретных устройств является величина потребляемой энергии схемы. Потребляемая мощность схемы, построенной на основе КМОП-технологии, пропорциональна интенсивности переключений узлов схемы. Это дает возможность частично решать данную проблему на уровне логического проектирования. Если не учитывать задержку сигналов в логических элементах, то повлиять на интенсивность переключений в фиксированной схеме невозможно, но использовать эту величину для снижения потребление энергии можно при покрытии ее библиотечными элементами. Покрывать надо так, чтобы узлы схемы с наибольшей интенсивностью переключений по возможности оказались внутри библиотечных элементов. Действительно, сокращение длины соединительных проводников в схеме ведет к уменьшению паразитных емкостей схемы, на перезарядку которых в процессе переключений расходуется энергия питания. В статье [Поттосин, 2011] предлагается метод покрытия логической схемы библиотечными элементами, основанный на сведении данной задачи к изоморфному вложению графов.

Заключение

Приведенный обзор не охватывает многочисленных работ по приложениям теории графов на всех этапах проектирования дискретных устройств, но показывает, насколько широко методы теории графов могут быть использованы для решения задач на всех стадиях логического проектирования. Большинство из рассмотренных задач теории графов представлено в курсе дискретной математики для студентов Белорусского государственного университета информатики и радиоэлектроники, специализирующихся в области проектирования дискретных устройств.

Библиографический список

[Агибалов, 1981] Агибалов, Г.П. Технология решения комбинаторно-логических задач методом сокращенного обхода дерева поиска / Г.П. Агибалов, В.А. Беляев. – Томск : Изд-во Томского ун-та, 1981. – 126 с.

[Бибило, 1987] Бибило, П.Н. Синтез комбинационных схем методами функциональной декомпозиции / П.Н. Бибило, С.В. Енин. – Минск : Наука и техника, 1987. – 189 с.

[Бибило, 1992] Бибило, П.Н. Синтез комбинационных ПЛМ-структур для СБИС / П.Н. Бибило. – Минск : Наука и техника, 1992. – 232 с.

[Закревский, 1981] Закревский, А.Д. Логический синтез каскадных схем / А.Д. Закревский. – М. : Наука, 1981. – 416 с.

[Закревский, 1986] Закревский, А.Д. Приложения теории графов к задачам логического проектирования дискретных устройств / А.Д. Закревский [и др.] // Исследования по

Библиотека БГУИР

прикладной теории графов. – Новосибирск : Наука, Сибирское отделение, 1986. – С. 3-9.

[Закревский, 2000] Закревский, А.Д. Раскраска графов при декомпозиции булевых функций / А.Д. Закревский // Логическое проектирование. – Минск : Ин-т техн. кибернетики НАН Беларуси, 2000. – Вып. 5. – С. 83-97.

[Закревский, 2005] Закревский, А.Д. Об оптимальном размещении графа в булевом пространстве / А.Д. Закревский // Вестник ТГУ. Приложение. – 2005. – № 14. – С. 13-17.

[Закревский и др., 2007] Закревский, А.Д. Логические основы проектирования дискретных устройств / А.Д. Закревский [и др.]. – М. : Физматлит, 2007. – 592 с.

[Мелихов, 1971] Мелихов, А.Н. Ориентированные графы и конечные автоматы / А.Н. Мелихов. – М. : Наука, 1971. – 416 с.

[Мелихов и др., 1974] Мелихов, А.Н. Применение графов для проектирования дискретных устройств / А.Н. Мелихов [и др.]. – М. : Наука, 1974. – 304 с.

[Нечепуренко и др., 1990] Нечепуренко, М.И. Алгоритмы и программы решения задач на графах и сетях / М.И. Нечепуренко [и др.]. – Новосибирск : Наука, 1990. – 515 с.

[Поттосин, 1985] Поттосин, Ю.В. Выделение максимальной двудольной части в неориентированном графе / Ю.В. Поттосин // Автоматизация решения логико-комбинаторных задач. – Минск : Ин-т техн. кибернетики АН БССР, 1985. – С. 22-28.

[Поттосин, 2001] Поттосин, Ю.В. Задачи теории графов в логическом проектировании / Ю.В. Поттосин // Логическое проектирование, вып.6. – Минск : Ин-т техн. кибернетики НАН Беларуси, 2001. – С. 106-130.

[Поттосин, 2011] Поттосин, Ю.В. Основы теории проектирования цифровых устройств / Ю.В. Поттосин. – Saarbrücken : LAP LAMBERT Academic Publishing, 2011. – 336 с.

[Armstrong, 1962] Armstrong, D.B. A programmed algorithm for assigning internal codes for sequential machines / D.B. Armstrong. – IRE Trans., EC–11, 1962. – N 4. – P. 466-472.

[Perkowski, 1995] Perkowski, M. A survey of literature on functional decomposition / M. Perkowski, S. Grygiel. – Portland State University, Department of Electrical Engineering, 1995 (Technical report). – 188 p.

THE PROBLEMS ON GRAPH IN THE THEORY OF DESIGN OF DIGITAL DEVICES

Pottosin Yu.V. *, Pottosina S.A. **

**United Institute of Informatics Problems, NAS of Belarus, Minsk, Republic of Belarus*

pott@newman.bas-net.by

*** Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus*

pottosina@sum-solution.net

The problems on graphs are considered, which have applications in the design of discrete control devices. A review of methods and algorithms for solving these problems is given. Examples of applications of graph models for solving logical design problems are given.

INTRODUCTION

The object-oriented approach to solving the problems of logical design of discrete devices supposes constructing a knowledge base. It contains structured information and inference rules that cover a given knowledge area by library elements. In logical design of discrete devices, such elements may be logical combinatorial problems and algorithms to solve them that are formulated in graph theory terms.

Graph theory methods are widely used in logical design of discrete control and computation devices. There are many works on applications graph theory in logical design and areas close to it, that are difficult to be enumerated. Therefore, we are restricted to mention some monographs [Мелихов, 1971], [Мелихов и др., 1974], [Нечепуренко и др., 1990] and reviews [Закревский и др., 1986], [Поттосин, 2001].

At the stage of logical design of discrete devices, various logical combinatorial problems arise [Агибалов, 1981], [Закревский и др., 2007], [Поттосин, 2011], whose solutions lead to optimal variants of logical projects. In this paper, we give a brief review of the methods and problems of graph theory that have wide application in logical design.

The concept of graph in the considered problems is used in different interpretations. A graph is often a graphic representation of a binary relation of compatibility, i.e. a relation having the properties of reflexivity and symmetry. Such a relation is represented by an undirected graph. In some cases, a graph represents a quasi-order relation or a consecution relation between some events. In some problems, a graph represents a logic network. The graphs are directed in these cases.

MAIN PART

The problem of graph coloring takes a considerable place in logical design. The graph coloring can be used in solving the problem of decomposition of Boolean functions that takes an important place in the synthesis of combinational circuits in VLSI basis. The problem of decomposition of Boolean functions may be set as follows. Given a system of Boolean functions as a vector function $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, and Boolean vector variables $\boldsymbol{w}$ and $\boldsymbol{z}$ composed of the components of vector variable $\boldsymbol{x}$. The given system of Boolean functions must be represented as $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{w}, \boldsymbol{g}(\boldsymbol{z}))$. A method to solve this problem supposes constructing graph G whose vertices are values $\boldsymbol{z}^*$ of $\boldsymbol{z}$, and there is an edge between $\boldsymbol{z}^*_i$ and $\boldsymbol{z}^*_j$ if and only if function $\boldsymbol{f}$ has different values at the same value of $\boldsymbol{w}$. It is proved that $\boldsymbol{f}$ can be represented as $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{w}, \boldsymbol{u})$ where $\boldsymbol{u}$ has k components if and only if the chromatic number of G is at most 2^k.

As it was said above, to solve the problem of decomposition of Boolean functions, one should encode the obtained colors by Boolean or ternary vectors. These vectors are the values of vector variable $\boldsymbol{u}$ that has the property that each edge of the graph must correspond to a component of $\boldsymbol{u}$ having opposite values (0 and 1) at the ends of the edge. So, a bipartite subgraph can be put in correspondence to each component of $\boldsymbol{u}$, and the problem can be considered as the problem of covering a graph by bipartite subgraphs. This problem arises in the search for optimal state assignment of a synchronous automaton. The specification of input and output is assumed to be in the structural alphabet. This model is called automaton with abstract state. An automaton is given by two ternary matrices, $\boldsymbol{U}$ and $\boldsymbol{V}$. The columns of $\boldsymbol{U}$ correspond to input Boolean variables $x_1, x_2, \ldots, x_n$ and the columns of $\boldsymbol{V}$ binary output signals $y_1, y_2, \ldots, y_m$. The rows of both matrices are marked by the states of the automaton so that a pair of their rows of the same

Библиотека БГУИР

name represents transition from the state, which is the mark of the row of $\boldsymbol{U}$, to the state, which is the mark of the row of $\boldsymbol{V}$. At that, the row of $\boldsymbol{U}$ represents the transition condition, and the row of $\boldsymbol{V}$ shows the output signals accompanying this transition. The method is following. Graph G is constructed for some column y_i of $\boldsymbol{V}$. The vertices of G correspond to the states of the given automaton. Pairs of elements with opposite values (0 and 1) are looked for in column y_i that correspond to pairs non-orthogonal rows of $\boldsymbol{U}$. The vertices of G corresponding to the marks of the rows of any such pair are connected with an edge and than are encoded as it is shown above. The obtained codes are put in the matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ as the values of the introduced internal variables $z_1, z_2, \ldots, z_k$. This procedure is repeated for every column of $\boldsymbol{V}$ and each time a column with minimum of pairs of mentioned elements is selected. When the graph G is empty (after introducing new variables the rows of $\boldsymbol{U}$ become orthogonal), the process finishes and the transformed matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ give the system of Boolean functions describing the combinational part of the logical circuit that implements the given automaton. Some compatible states can be encoded by non-orthogonal or the same codes. In this case the number of states of the automaton becomes reduced.

The problems of logical design formulated as problems of partitioning a given set into compatible subsets in a certain sense can be reduced to graph coloring if pair-wise compatibility of the set elements is enough for their group compatibility. Otherwise, one should introduce some additional conditions for the elements to be of the same color. Another approach to solving such problems assumes finding in a given graph the largest independent set, i.e. the largest set of pair-wise nonadjacent vertices, and then grouping at them the rest of vertices of the graph. This approach can be used to solve such problems as minimization of code length of states of an asynchronous automaton, compaction of the flow table of an automaton (by rows and columns), parallel decomposition of an automaton, simplification of the system of Boolean functions describing an asynchronous automaton. This technique suits especially for solving such problems of abstract synthesis of automata as state minimization and decomposition of a synchronous automaton. In these problems, the desired family of compatible sets must be closed, i.e. existence of some elements in a compatible subset demands including some other elements in another compatible subset belonging the same family. Finding complete subgraphs is a dual problem with respect to finding independent sets, because a complete subgraph of a graph is induced by an independent set of the complement of the graph. To find all maximal independent sets the classical Paull-Unger method for finding all maximal compatible sets of states of an automaton can de used. It uses a technique where the given graph is decomposed into the sequence of subgraphs, and the solution for preceding subgraph is transformed into the solution for the next subgraph. Another algorithm for finding all maximal independent sets is suggested that uses the lexicographical search.

The model of complete Boolean graph Q_k is used to solve the problem of state assignment of discrete automata that must be solved to transit from behavior description of an automaton to structural one. The optimization criterion is the chip area or power consumption. The process of state assignment can be considered as assembling a graph Q_k whose vertices are Boolean vectors (elements of Boolean space). The process of constructing the graph Q_k, which is called k-dimensional hyper-cube, is a sequence of k steps. At the p-th step, the set of $(p-1)$-dimensional hyper-cubes (graphs Q_{p-1}) is considered. They are joint in pairs, and adding edges transforms each pair into one p-dimensional hyper-cube (graph Q_p). At that, as far as it is possible, those pairs of vertices are chosen for connecting with edges, which maximal values of some integral function w_{ij} determined at the pairs of states correspond to. The vertices of the obtained graph Q_k are assigned k-component Boolean vectors keeping neighbor relation between the vertices represented by the edges of graph Q_k.

Rather considerable variety of graph problems is in optimization of the structures of very large scale integration (VLSI). The problems of optimization of VLSI structures are at the turn of logical and technological designs. In the process of optimization of the structure of a programmable logical array (PLA), which is one of VLSI structures, the problem of folding PLA, i.e. using the same line (horizontal or vertical) to embed different wires. This results in reducing the chip area. In [Бибило, 1992], finding a multiple folding of a PLA, when more than two wires are embedded on the same line and outer connection is carried to "the third dimension", is reduced to graph coloring. The vertices of the graph correspond to the wires and two vertices are adjacent if and only if the correspondent wires cannot be embedded on the same line. The graph coloring indicates the way of wire embedding. Given a partition of the set of external signals into two subsets corresponding to the signals supplied to the PLA from different sides, the problem of finding the maximal number of folded pairs is reduced to the problem of finding the maximal matching in a bipartite graph. The parts of this graph correspond to the mentioned subsets, and there is an edge between two vertices of different parts if the corresponded signals form a pair of folding.

CONCLUSION

This review does not include many works on applications of graph theory at all the stages of designing digital devices, but it shows how wide the graph theory methods can be used in logical design. The majority of the considered problems are present in the course of discrete mathematics for students of Byelorussian State University of Informatics and Radio-Electronics.

Библиотека БГУИР