УДК 004.023 DOI: 10.31114/2078-7707-2020-4-95-102

О сложности инверсных графов, реализующих булевы функции от малого числа переменных

С.А. Ложкин, В.С. Зизов, М.С. Шуплецов,

В.В. Жуков, Д.Э. Хзмалян, О.О. Белянков

Московский государственный университет имени М.В. Ломоносова – факультет вычислительной математики и кибернетики, lozhkin@cs.msu.ru

Аннотация — **В работе рассматривается задача построения каталогов схем, реализующих функции алгебры логики от малого числа переменных. Данная задача рассматривается для двух моделей схем из функциональных элементов, описанных в тексте. Общей чертой моделей является наличие элемента отрицания нулевой стоимости, различаются модели своими базисами. Для решения задачи разработаны алгоритмы синтеза схем, и на их основе реализованы программные инструменты. С применением описанных методов получены минимальные и близкие к ним схемы для всех функций алгебры логики от пяти переменных. Установлена верхняя оценка сложности реализации функций в этих моделях. Показана верхняя оценка средней сложности схем в этих моделях.**

Ключевые слова — **методы синтеза схем, схема из функциональных элементов, базис Поста, сложность булевых функций, средняя схемная сложность, функция голосования, база данных схем.**

I. Введение

Одной из актуальных задач математической кибернетики является построение оптимальных схем в заданной модели дискретных управляющих систем. Отдельное место занимают каталоги или библиотеки схем, содержащие реализации функций малого числа переменных. Такие базы данных находят применение в различных алгоритмах логического синтеза цифровых схем.

Алгоритмы синтеза схем хорошо исследованы и применяются в разных областях. Большинство таких методов дают результат, удовлетворительный для промышленного производства [1]. Однако наибольший интерес представляет точный синтез: получение минимальных схем в некотором смысле. Обычно задача формулируется как минимизация функционала качества, который может представлять собой комбинацию нескольких характеристик. Схема из функциональных элементов (СФЭ) может пониматься как математическая модель физического устройства электрической схемы. Для этих схем естественным образом возникают такие показатели, как число элементов электрической схемы, задержка, энергопотребление и так далее. СФЭ, как математическая абстракция, лишена ряда ограничений, накладываемых на электрические схемы.

Одним из основных функционалов качества таких схем остаётся число элементов. Важной задачей является задача точного синтеза минимальных схем в некоторой заданной модели. В связи с высокой временной и вычислительной сложностью такие методы синтеза применяются очень ограниченно. Обычно их применение невозможно для функций большой размерности.

Целью данной работы является описание и анализ различных методов получения минимальных по сложности и близких к ним схем. В работе рассматриваются методы синтеза схем для функций алгебры логики (ФАЛ) малой размерности, не более чем 5 переменных. Актуальность задачи заключается в том, что методы синтеза более сложных ФАЛ большого числа переменных, часто основаны на разложении или декомпозиции задачи и сведению её к построению схем для функций малого числа переменных. Таким образом, улучшив результаты и характеристики схем для ФАЛ малого числа переменных, можно получить лучшие результаты для всех других ФАЛ.

Для достижения данной цели поставлены задачи описания различных методов синтеза схем, в частности методов, направленных на получение минимальных схем, и реализации части алгоритмов для получения минимальных схем, а также описания качественных результатов, включая известные верхние оценки сложности.

Среди моделей логического синтеза можно выделить модели, обладающие общими чертами. Конъюнктивно-инверсный граф (And-Inverter Graph, AIG) является специальным классом СФЭ в базисе из элемента конъюнкции и элемента отрицания. При этом сами элементы отрицания имеют нулевой вес и представлены специальными пометками на ребрах графа. Абстрактный граф AIG может быть представлен в виде эффективной структуры данных. Преобразование из сети логических вентилей в AIG происходит быстро и масштабируемо. Для этого требуется лишь, чтобы каждый элемент был выражен в терминах элементов конъюнкции и отрицания. Это преобразование линейно как по памяти, так и по времени. Это делает данную модель эффективным представлением по сравнению с бинарной диаграммой решений (Binary Decision Diagram, BDD).

Интерес к AIG начался с основополагающей работы А. Тьюринга 1948 года о нейронных сетях, в которой он описал обучаемую сеть NAND вентилей. Работа в данном направлении продолжалась вплоть до конца 1950-х годов. Например, Л. Хеллерманом в 1963 году была опубликована работа [6]. В ней был составлен и описан каталог схем для всех ФАЛ от 3 переменных. В 1970-е годы были разработаны различные локальные преобразования. Эти преобразования были реализованы в нескольких системах логического синтеза и верификации, которые, например, описаны в работах Даррингера [3]. Они уменьшают схемы для улучшения площади и задержки при синтезе или для ускорения проверки формальной эквивалентности.

В последнее время возрос интерес к AIG как функциональному представлению для различных задач синтеза и верификации. Важным событием стало появление гораздо более эффективных программ для решения задач выполнимости булевых формул (boolean satisfiability problem, SAT). Особенно популярными они становятся из-за хорошей сочетаемости с современными вычислительными архитектурами ЭВМ [5]. Модель нашла успешное применение в различных приложениях автоматизации проектирования электроники (EDA) [8]. Хорошо настроенная и отлаженная комбинация AIG и SAT оказала влияние на формальную верификацию, включая как верификацию моделей, так и проверку эквивалентности схем. Работы П.Бюссе и А. Боралва [2] дают понимание того, что задачи логического и физического синтеза могут быть решены с помощью моделирования в терминах выполнимости логических формул для вычисления функциональных свойств (таких как симметрии). А. Мищенко с соавторами [7] показали, что AIG являются перспективным представлением, которое может объединить логический синтез, технологическое отображение, физический синтез и формальную верификацию. Их работа нашла выражение в виде программного пакета ABC, основным разработчиком которого является А. Мищенко.

Мажоритарно-инверсный граф (Majority-Inverter Graph, MIG) является относительно новой логической структурой представления для эффективного синтеза оптимизации булевых функций. Такая схема представляет специальный класс СФЭ в базисе из элемента функции голосования трех переменных, константы и элемента отрицания. Как показали Л.Амару с соавторами [1], MIG включают в себя AIG. Чтобы поддержать естественную обработку MIG, была рассмотрена булева алгебра с полной аксиоматической системой. Экспериментальные результаты, полученные с помощью пакета mcnc benchmark suite, показали, что оптимизация MIG уменьшает использование логических элементов после физического синтеза в среднем на 18% по сравнению с оптимизацией AIG, выполняемой ABC.

II. Основные определения

Введём несколько определений, которыми будем пользоваться в работе. Всюду далее будем понимать под функциями - функции алгебры логики (ФАЛ). Разобьем пространство функций на множество классов эквивалентности таким образом, чтобы любая функция из такого класса была равна другой функции того же класса с точностью до инверсии и перестановки переменных либо инвертированию вектора значений функции. Назовём их инверсно-конгруэнтными классами (ИКК, далее класс). СФЭ реализует класс, если она реализует какую-то функцию этого класса.

III. Описание моделей

A. *Инверсные графы*

Инверсный граф – это ориентированный ациклический граф, имеющий пометки на рёбрах и вершинах, и ограничение на степень захода вершин. Каждая вершина, степень захода которой равна 0, называется входом. Все прочие вершины будем называть внутренними вершинами. Выделим среди них множество вершин, называемое выходами. Будем говорить, что такая схема задана своим базисом, то есть множеством допустимых функциональных элементов.

Пометки на рёбрах означают отрицание на соответствующем входе функционального элемента, для которого данное ребро является входящим. Отсутствие пометки означает отсутствие отрицания. Будем далее называть такие состояния рёбер, или их пометки, противоположными. Каждому входу графа приписывается метка входа, которая является булевой переменной либо константой. Всем внутренним вершинам сопоставлена своя пометка одной из функций базиса. Ограничение на степень входа вершины является существенным для конечных базисов. Для каждой вершины, помеченной n -входовым функциональным элементом, степень захода равна n. Каждому выходу приписывается уникальная метка выхода и, возможно, пометка, указывающая на отрицание выхода схемы.

Схема реализует одну или несколько булевых функций, если на выходах СФЭ реализуются эти булевы функции от переменных, являющихся входами схемы. Схемы, реализующие на выходах одинаковое множество функций, называются эквивалентными. Отметим, что функции базиса должны составлять в совокупности с элементом отрицания и, возможно, константами, полный базис [12]. Схемной сложностью такой СФЭ будем считать количество функциональных элементов, или, что то же самое, число внутренних вершин. Будем далее понимать под сложностью именно данный функционал. Сложностью в среднем случае для функций от n переменных будем называть сложность, среднюю для множества всех булевых функций от n либо менее переменных. Опишем далее несколько моделей, отличающихся базисами и, соответственно, ограничениями на степень захода вершин.

B. *Базис Поста*

Базис Поста состоит из элементов конъюнкции и отрицания. В англоязычной литературе модель обычно называется And-Inverter Graph, AIG. Это модель формализуется как инверсный граф, где каждая внутренняя вершина имеет степень захода 2. Внутренним вершинам графа соответствуют только пометки “&”, обозначающие элемент конъюнкции. С использованием отрицаний на рёбрах можно смоделировать функции дизъюнкции и отрицания. Все входы графа соответствуют булевым переменным.

Рассмотрим для примера функцию «XOR», или сумму по модулю два двух переменных. Она же исключающее «ИЛИ». Такая функция может быть реализована разными способами, но не менее, чем в три функциональных элемента. Рассмотрим схемы в данном базисе. Здесь и далее пометкой “&” обозначается элемент конъюнкции, зелёным (светло-серым) ребром прямое соединение, красным (тёмно-серым) ребром отрицание. Для определённости те иллюстрации, где выход инвертирован, содержат соответствующую запись в подписи к рисунку.

Первый способ - выразить функцию через совершенную дизъюнктивную нормальную форму, см. рис. 1. Другой способ - реализовать ту же функцию, представив её в виде конъюнкции «ИЛИ» и отрицания «И», см. рис. 2.

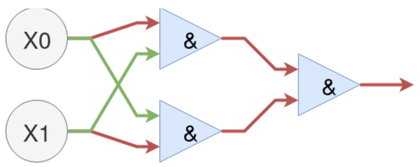

Рис. 1. СФЭ в модели AIG, реализующая функцию «исключающее ИЛИ» (выход инвертирован)

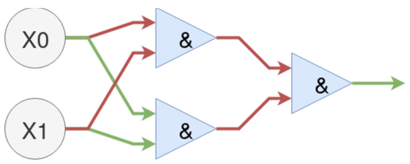

Рис. 2. Альтернативная СФЭ в модели AIG, реализующая функцию «исключающее ИЛИ»

C. Базис функции голосования

Второй моделью схем из функциональных элементов является модель, основанная на функции голосования трёх переменных. В англоязычной литературе модель появилась относительно недавно, и называется Majority-Inverter Graph, MIG. Впервые она была представлена в публикации в 2014 г. [1]. Она задана базисом из функции голосования трёх переменных и отрицания. Этот базис представляет собой полный базис, если допускается использование константы, нуля либо единицы. Такая модель нашла свое приложение в синтезе схем из библиотеки более сложных элементов, у которых число входов превышает три [9].

Модель формализуется как инверсный граф, у которого внутренним вершинам всегда сопоставляется пометка “M”, обозначающая функциональный элемент голосования 3-х переменных. Степень входа всех внутренних вершин равна 3. Входами схемы могут быть булевы переменные либо константа 0.

Утверждается, что любая схема в модели AIG может быть сопоставлена схеме в базисе из функции голосования той же сложности. Способ сопоставления заключается в замене каждого элемента конъюнкции на элемент голосования, в котором третье ребро соединено с константой 0.

В качестве иллюстрации приведём схему, реализующую ту же функцию суммы по модулю два в модели MIG (рис. 3). Здесь пометкой “M” обозначается элемент голосования.

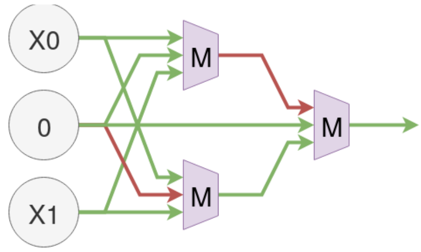

Рис. 3. Минимальная СФЭ в модели MIG, реализующая функцию «исключающее ИЛИ»

Было выдвинуто предположение, что схемная сложность для реализации произвольной ФАЛ в модели MIG в среднем существенно меньше, чем в модели AIG. Это продиктовано тем соображением, что число входов функциональных элементов больше, чем в предыдущей модели, а множество схем является своего рода надмножеством схем в предыдущей модели.

IV. Обзор алгоритмов синтеза

A. Точный синтез через перебор неизоморфных графов

Алгоритмы точного построения схем представляют собой алгоритмы синтеза, направленные на получение для заданной булевой функции схемы, отвечающей определенным условиям. В задаче построения минимальных схем таким условием будет являться минимальность сложности схемы среди множества других схем, реализующих ту же функцию.

Показал себя перспективным метод точного синтеза схем малой сложности путём последовательного перебора всех неизоморфных ориентированных графов. Он заключается в генерации всевозможных матриц инцидентности, отвечающих условиям модели. Каждая такая матрица задаёт ориентированный ациклический граф, уже отвечающий ограничениям на степень захода вершин. Для получения схемы необходимо лишь разметить рёбра, так как разметка вершин может быть произведена в процессе построения графа.

Будем считать, что в каждой вершине итоговой схемы реализуется некоторая ФАЛ. Допустим, в заданный момент времени имеется ориентированный ациклический граф, где в каждой вершине хранится отображение из множества пометок на рёбрах графа в множество функций, которые реализуются в данной вершине. Будем говорить, что в таких вершинах реализовано это множество функций для заданного строения графа. Также будем считать, что это состояние представлено в битовом векторе в памяти компьютера. Добавим еще одну вершину. Она увеличила сложность схемы на единицу, и реализует большее число функций.

Зафиксируем p вершин, которые являются истоками для входов функционального элемента. Тогда число элементов множества всевозможных разметок входных рёбер равно 2^p. В каждой из этих вершин реализуется некоторое множество функций. Тогда в новой вершине реализуется число функций не более, чем произведение числа элементов этих множеств, умноженное на 2^p. Более точно, для m полученных ранее разметок рёбер графа, новое число разметок есть $m2^p$, и ровно такое же число функций, возможно, тождественно равных.

Структура ориентированного графа без каких-либо пометок на рёбрах является уникальной. Любой граф, изоморфный этому, будет реализовывать то же множество ФАЛ. Закончив построение этого множества, возможно вернуться на шаг назад, к графу на единицу меньшей сложности. Нет необходимости заново рассчитывать множество функций, реализуемых в его вершинах, если допустимо хранить состояния в памяти ЭВМ. Более того, отображение множества пометок в множество реализуемых в вершине функций можно не менять при добавлении новых вершин. Это обусловлено тем, что любая новая разметка получается добавлением ребра с пометкой, что не влияет на реализуемые в вершине функции. Значит, существует сюръективное отображение новых разметок в старые. Такое отображение строится путём отбрасывания новых пометок.

Приведём пример схемы, реализующей сумму трёх переменных по модулю два, которая имеет сложность три в модели MIG, см. рис 4. Данная схема получена описанным методом, и является минимальной.

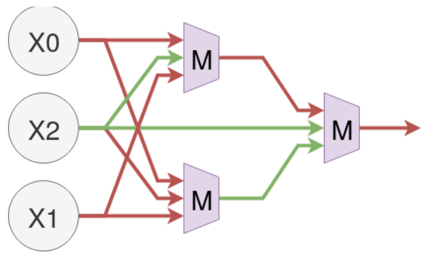

Рис. 4. Минимальная схема из функциональных элементов голосования, реализующая ФАЛ суммы по модулю два от 3 переменных (выход инвертирован)

Данный метод позволяет получить оптимальные в смысле сложности схемы для некоторого множества ФАЛ. С использованием структурного хеширования можно значительно ускорить проверку изоморфизма графов. В модели MIG мы получили все оптимальные схемы для всех функций, реализуемых схемой сложности четыре и менее, и некоторое число для функций, реализуемых схемой из пяти внутренних вершин.

Достоинствами данного метода являются реализация оптимальных схем для ФАЛ и, более того, реализация всех оптимальных схем для каждой ФАЛ. Иногда это является критически важным. Если задача состоит в оптимизации большой ФАЛ, то можно отдать предпочтение одной из нескольких схем. В таком случае может быть выбрана схема, наилучшим образом подходящая к конкретным условиям. Например, в вершинах которой реализуются функции, необходимые для других схем, или наименьшая по задержке.

Недостатком метода является высокая пространственная сложность алгоритма. Несмотря на возможные оптимизации, требуется хранить сведения обо всех неизоморфных графах.

B. Точный синтез через формулирование задачи выполнимости КНФ

Ранее в работе [10] были получены оптимальные схемы в модели MIG для функций от четырёх переменных. Данные результаты были получены с применением SAT-решателей (SAT-solver). Опишем подробнее сведение задачи синтеза схемы к задаче выполнимости конъюнктивной нормальной формы (КНФ) в случае базиса модели из функции голосования. Будем выражать задачу в терминах элементарных функций с тем, чтобы автоматическое преобразование могло дать эквивалентную КНФ. Сформулируем задачу "существует ли схема сложности K, реализующая заданную функцию?". Для этого запишем все логические узлы в виде некоторой формулы. Например, для модели MIG логическим узлом является функция голосования Maj трёх переменных. Соответствующая формула примет, при входных переменных x_1, x_2, x_3,вид

$$Maj(x_1, x_2, x_3) \overset{\text{def}}{=} x_1\mathrm{x}_2 \vee \mathrm{x}_2\mathrm{x}_3 \vee \mathrm{x}_1\mathrm{x}_3.$$

Все соединения между узлами можно выразить в виде

$$(\mathrm{in_i} \equiv \mathrm{out_1}) \wedge \mathrm{connect}_{\mathrm{io_1}} \vee \dots (\mathrm{in_i} \equiv \mathrm{out_{i-1}}) \wedge \mathrm{connect}_{\mathrm{io_{i-1}}}$$
$$\vee (\mathrm{in_i} \equiv \mathrm{x_1}) \wedge \mathrm{connect}_{\mathrm{ix_1}} \vee \dots (\mathrm{in_i} \equiv \mathrm{x_n}) \wedge \mathrm{connect}_{\mathrm{ix_n}},$$

где формулами связаны выходы всех предыдущих по топографическому порядку узлов и входы (каждый $\mathrm{in_i}$ отдельно) текущего. Отдельное условие $\wedge\,\mathrm{connect_{ij}}$ означает в конфигурации схемы подключение контакта i к контакту j, в данном случае - входа in_i логического узла к контакту j схемы. Необходимо дополнительно ограничивать множество решений условием, что лишь одно соединение $connect_{ij}$ может быть истинным для фиксированного i. Каждый вход схемы может быть инвертирован, что учитывается отдельным контактом

$$(\mathrm{x_1} \equiv \mathrm{in_1}) \wedge \neg\mathrm{inverter_1} \vee (\mathrm{x_1} \equiv \neg\mathrm{in_1}) \wedge \mathrm{inverter_1}$$

в формульном описании логического элемента.

Выход последнего узла out_k полагается выходом схемы, и должен быть равен заданной функции out, либо её отрицанию в случае инвертирования выхода.

$$(\mathrm{out} \equiv \mathrm{out_k}) \wedge \neg\mathrm{inverter_{out}} \vee$$
$$(\mathrm{out} \equiv \neg\mathrm{out_k}) \wedge \mathrm{inverter_{out}}.$$

Такая постановка позволяет сформулировать задачу для единственного набора $\mathrm{x_1}, \dots, \mathrm{x_n}$. Вся структура схемы исчерпывающе описывается набором переменных-соединений и инвертеров. Для того, чтобы сформулировать задачу выполнимости КНФ для схемы на всех наборах, достаточно сформулировать задачу для каждого набора с общими переменными соединений и инвертеров. Практически это выражается в связывании логических переменных x_i с вектором значений. Например, для двух переменных x_1, x_2 корректным будет использовать $\mathrm{x}_1 = 0101, \mathrm{x}_2 = 0011$ на входах схемы. Данная форма записи в SAT-решателях, поддерживающих формат решения bit-vec, переписывается в виде векторов значений в шестнадцатеричном формате, и задача выполнимости проверяется одновременно для всех наборов.

Аналогичные рассуждения применимы и для модели AIG. Описанные выражения можно автоматически преобразовать в КНФ. Например, через преобразование Цейтина, описанное в [13].

В оригинальной работе СФЭ сводились к задаче выполнимости КНФ. При таких условиях возможно для каждой ФАЛ f найти такое минимальное k, что f реализуется схемой сложности k. Реализация данного подхода заняла несколько тысяч машино-часов. Было признано целесообразным получить схемы для ФАЛ, имеющих наибольшую сложность реализации. Тогда, если известна верхняя оценка сложности для функции f в L вершин, то задача состоит в нахождении схемы на единицу меньшей сложности.

Данный метод позволяет найти оптимальные по сложности схемы для каждой функции отдельно. Основная сложность данного подхода заключается в работе SAT-решателя. От выбора конкретной программной реализации значительно зависит скорость решения. Некоторые эвристические заключения помогают несколько сократить размерность пространства решений и упростить поиск схемы. Например, наблюдения, касающиеся порядка контактов или частичного порядка вершин.

Достоинством метода является оптимальность полученных схем, либо формальное доказательство невозможности построения для заданной функции схемы требуемой сложности. Он может применяться для получения нижних оценок.

Недостатком метода является высокая временная сложность алгоритма, прямо связанная со сложностью решения задачи выполнимости КНФ.

C. Сокращение синтезированных схем

Задача логического синтеза заключается в составлении СФЭ, реализующих функции алгебры логики [4]. Условно этот процесс можно разделить на две части. Первая подзадача заключается в синтезе схемы, а вторая в ее оптимизации. Алгоритмы минимизации заданной схемы ориентированы на синтез схемы для одной функции каким-либо произвольным методом и её сокращение применением эквивалентных преобразований к подсхемам. В англоязычной литературе такие алгоритмы называются алгоритмами re-writing.

В случае составления каталога схем наиболее простым выглядит его последовательное построение с использованием разных методов. Сначала алгоритмами точного синтеза находятся схемы небольшой сложности и схемы для функций от меньшего числа переменных. Затем для всех функций синтезируется хотя бы по одной схеме простым методом, например, разложением Шеннона или методом каскадов. Здесь не фиксирован порядок переменных, а значит, для каждой функции существует множество эквивалентных схем. Все такие схемы можно последовательно упрощать, применяя, например, замену подсхем на эквивалентные, полученные первым методом (A).

Преимущество таких подходов состоит в том, что они относительно просты в использовании, и характеризуются низкими временными затратами. Основная область их применения – синтез схем для функций от большого числа переменных. Для таких методов является критически важным условие наличие базы данных эквивалентных схем для функций от малого числа переменных. Результаты данной работы могут служить для построения такой базы.

Недостатками данного семейства методов является их сильная зависимость от начальной базы данных эквивалентных схем и локальность вносимых изменений. Большим недостатком можно считать то, что в общем случае они не дают минимальной схемы, а позволяют лишь уменьшить сложность схемы до приемлемых значений.

D. Итеративные методы построения каталога схем

Методы сокращения уже имеющихся схем могут найти только некоторые оптимизации схем, уменьшить их сложность в константу раз, но не на порядок роста. Последний зависит только от начального алгоритма синтеза. Такие методы полагаются на некоторую базу соответствующих эквивалентных схем. Можно составить каталог таких оптимальных схем, охватывающий большое число функций.

Возникла идея использовать итеративный алгоритм нахождения множества схем, близких к минимальным. Его суть состоит в составлении каталога неоптимальных схем и их постепенного улучшения. Алгоритмы минимизации множества схем представляют собой подход, ранее использовавшийся в работе [11]. Такие алгоритмы направлены на уменьшение средней сложности схем в некоторой базе данных. В них применяются неэквивалентные преобразования, приводящие к получению новых схем, реализующих, возможно, другие булевы функции. Результатом повторения нескольких итераций становится снижение средней сложности схем за счёт нахождения схем меньшей сложности для некоторых функций.

Опишем это более подробно. Возьмем некоторую схему, имеющую сложность k. Путём добавления либо удаления функционального элемента может быть получена схема, реализующая иную ФАЛ. Такая схема может иметь сложность $n-1, n, n+1$. В процессе работы для каждой ФАЛ накапливаются реализующие её схемы. Среди таких схем не представляют интереса схемы большой сложности. Наоборот, схемы минимальной сложности могут быть зафиксированы и представлены в виде каталога схем, являющегося результатом работы алгоритма. Такой каталог состоит из одной схемы для каждого ИКК.

Главным параметром метода является набор преобразований, или эвристик, которые в нём применяются. Если задан начальный каталог неоптимальных схем, то после каждого шага будут получены все схемы, которые возможно получить одним преобразованием такого вида. После повторения некоторого числа шагов, или итераций, будет получено множество схем, которые возможно получить за это число неэквивалентных преобразований. Таким способом любая оптимальная схема может быть получена из некоторой неоптимальной.

Существенно важной в методе является проверка того, что полученная схема меньше, чем имеющаяся в базе данных реализация той же функции. Если полученная схема отвечает данному условию, то она замещает соответствующую запись, и к ней снова применяются те же преобразования. Любая схема, полученная в результате такого перебора, даёт верхнюю оценку сложности реализуемой функции алгебры логики. После окончания работы нельзя утверждать, что получены все минимальные схемы. Это связано с тем, что не все схемы достижимы такими однократными преобразованиями из исходного набора схем.

Был опробован ряд эвристик. Добавление или удаление одного функционального элемента описано выше. Кроме этого, использовалось изменение пометки отрицания на ребре на противоположную. Данная эвристика применялась для всевозможных рёбер. Ещё двумя типами преобразований являются перемещение ребра и перемещение выхода схемы.

Основными проблемами, которые решаются при разработке программных систем на основе данного метода, являются следующие: хранение базы данных всех схем; получение новых, и критерий принятия схемы либо отказа от обработки; симуляция, и проверка реализуемого ИКК.

Данный метод позволяет найти близкие к минимальным схемы для множества заданных функций. Описанный алгоритм занимает промежуточное состояние по временной и пространственной сложности между описанными ранее алгоритмами.

Достоинствами данного подхода можно считать меньшее, чем в алгоритмах точного синтеза, соотношение временной и пространственной сложностей. По сравнению с методами сокращения схем, данный подход даёт более близкие к минимальным схемы.

Недостатками этого подхода является его ограничение на число схем, являющихся представителями для каждого ИКК. Также существенным недостатком является сильная зависимость от начальной базы схем, и набора применяемых преобразований.

V. Полученные результаты

По результатам численного эксперимента была сформирована база данных, или каталог, схем. Покажем две схемы, каждая из которых является минимальной в своей модели. Обе они реализуют сумму трёх переменных по модулю два, см. рис. 4, 5.

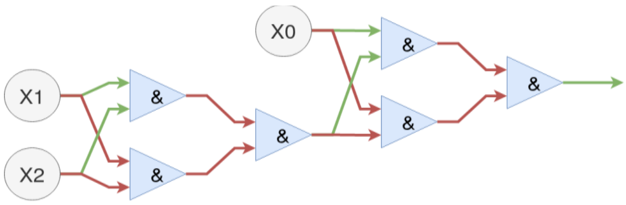

Рис. 5. Минимальная схема в модели AIG, реализующая сумму по модулю два от 3 переменных

База данных схем имеет распределение сложностей схем в модели MIG, реализующих функции от 5 переменных, которое показано на рис. 6. Верхняя оценка сложности равна 11. Это ровно один ИКК, и есть вероятность, что в дальнейших работах получится улучшить реализующую его минимальную схему, см. рис. 8. Средняя сложность схем для всех функций от 5 переменных составляет 7,701, см. рис. 6. Данная оценка является верхней оценкой.

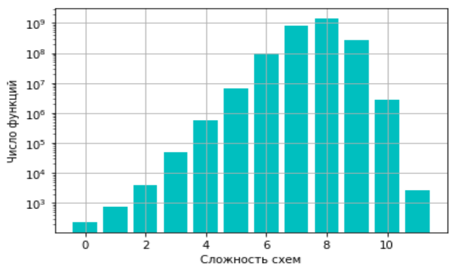

Рис. 6. Число ФАЛ от 5 переменных в зависимости от верхней оценки сложности их минимальной реализации в модели MIG, в логарифмической шкале

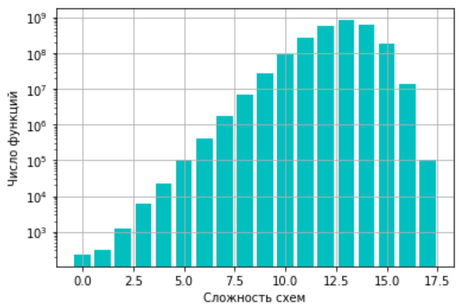

Рис. 7. Число ФАЛ от 5 переменных в зависимости от верхней оценки сложности их минимальной реализации в модели AIG, в логарифмической шкале

Распределение сложностей схем, реализующих ФАЛ от 5 переменных в модели AIG показано на рис. 7. Аналогичная верхняя оценка средней сложности схем в этом базисе составила 12,800. Верхняя оценка сложности равна 17. В полученном каталоге схемы в модели MIG в среднем имеют сложность на 39,8% меньше, чем в модели AIG.

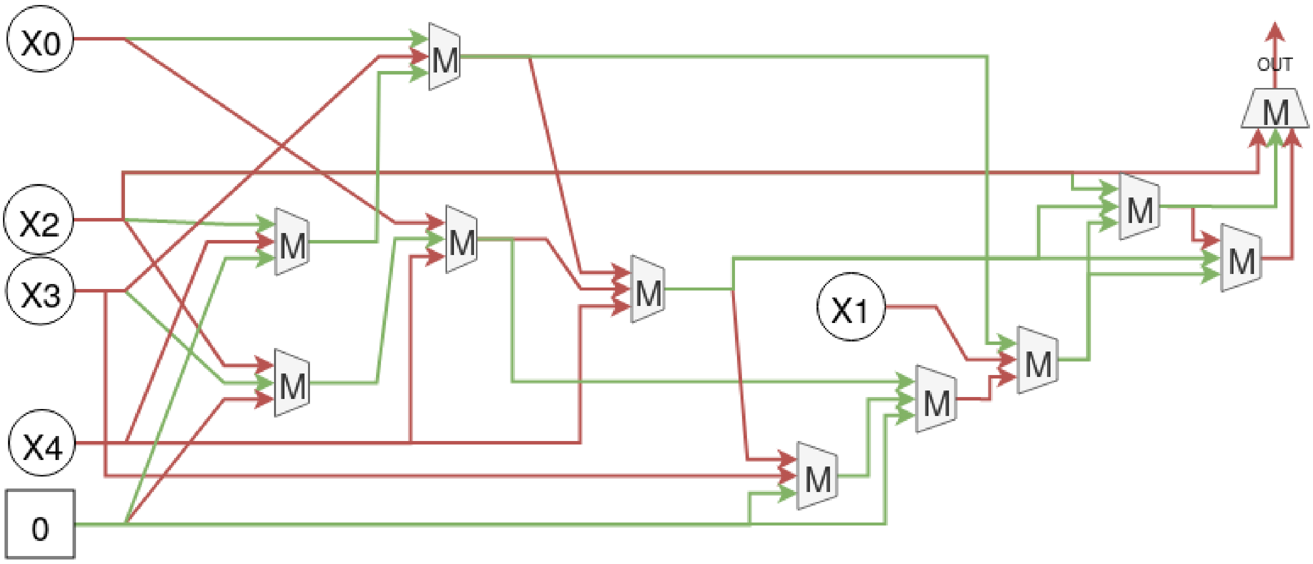

Рис. 8. СФЭ голосования, соответствующая единственному ИКК сложности 11. Реализует булеву функцию с вектором значений 0001 0110 1010 1001 1100 1001 0111 1000, выход инвертирован

VI. Заключение

В настоящей работе описаны различные алгоритмы точного синтеза схем, приведены описания новых перспективных алгоритмов синтеза. Данные алгоритмы применимы к любой модели инверсных графов. Проведено качественное сравнение методов и подходов. На основании проведённой работы построены программные реализации описанных подходов и составлен каталог схем из функциональных элементов в двух рассмотренных базисах. Показаны верхние оценки сложности схем в моделях AIG и MIG и верхние оценки средней сложности схем в этих моделях. Показано распределение данных сложностей.

Поддержка

Работа выполнена при финансовой поддержке РФФИ – грант № 18-01-00800-а.

Литература

[1] Luca Amaru´, Pierre-Emmanuel Gaillardon и Giovanni Micheli. «Majority-Inverter Graph: A Novel Data-Structure and Algorithms for Efficient Logic Optimization». // Proceedings of the 51st Design Automation Conference (DAC) (июнь 2014). DOI: 10.1145/2593069.2593158.

[2] Per Bjesse и Arne Boralv. «DAG-aware circuit compression for formal verification» дек. 2004, с. 42—49. ISBN: 0-7803-8702-3. DOI: 0.1109/ICCAD.2004.1382541.

[3] John Darringer, William Jr, C. Berman и Louise Trevillyan. «Logic Synthesis Through Local Transformations». // IBM Journal of Research and Development 25 (июль 1981), с. 272—280. DOI: 10.1147/rd.254.0272.

[4] Kanupriya Gulati. Advanced Techniques in Logic Synthesis, Optimizations and Applications. янв. 2011. ISBN: 978-1-4419-7517-1. DOI: 10.1007/978-1-4419-7518-8.

[5] Winston Haaswijk, Mathias Soeken, Alan Mishchenko и Giovanni Micheli. «SAT- Based Exact Synthesis: Encodings, Topology Families, and Parallelism». // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (февр. 2019), DOI: 10.1109/TCAD.2019.2897703.

[6] Leo Hellerman. «A Catalog of Three-Variable Or-Invert and And-Invert Logical Circuits». // Electronic Computers, IEEE Transactions on EC-12 (июль 1963), с. 198 —223. DOI: 10.1109/PGEC.1963.263531.

[7] Heinz Riener, Winston Haaswijk, Alan Mishchenko, Giovanni De Micheli и Mathias Soeken. «On-the-fly and DAG-aware: Rewriting Boolean Networks with Exact Synthesis». // 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE) (2019), с. 1649—1654.

[8] L. Scheffer, L. Lavagno и G. Martin. EDA for IC implementation, circuit design, and process technology. янв. 2006, с. 1—596.

[9] Mathias Soeken, Luca Amaru, Pierre-Emmanuel Gaillardon и Giovanni Micheli. «Exact Synthesis of Majority-Inverter Graphs and Its Applications». // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (февр. 2017), DOI: 10.1109/TCAD.2017.2664059.

[10] Mathias Soeken, Luca Gaetano Amaru`, Pierre-Emmanuel Gaillardon и Giovanni De Micheli. «Optimizing Majority-Inverter Graphs with Functional Hashing». // Proceedings of the 2016 Conference on Design, Automation & Test in Europe. DATE ’16. Dresden, Germany: EDA Consortium, 2016, с. 1030–1035. ISBN: 9783981537062.

[11] С. А. Ложкин, М. С. Шуплецов, В. А. Коноводов, Б. Р. Данилов, В. В. Жуков и Н. Ю. Багров. «Распределенная система и алгоритмы поиска минимальных и близких к ним контактных схем для булевых функций от малого числа переменных». // Сборник трудов Всероссийских научно-технических конференций "Проблемы разработки перспективных микро и наноэлектронных систем (МЭС)" 1 (2016), с. 40—47.

[12] С. С. Марченков. Замкнутые классы булевых функций. ФИЗМАТЛИТ Москва, 2000. ISBN: 5-9221-0066-1.

[13] Г. С. Цейтин, «О сложности вывода в исчислении высказываний», Исследования по конструктивной математике и математической логике. II, Зап. научн. сем. ЛОМИ, 8, Изд-во "Наука", Ленинград. отд., Л., 1968, с. 234–259

On Complexity of Inverter Graphs for Boolean Functions of Small Number of Variables

S.A. Lozhkin, V.S. Zizov, M.S. Shupletsov,

V.V. Zhukov, D.E. Khzmalian, O.O. Belyankov

Lomonosov Moscow State University – Faculty of Computational Mathematics and Cybernetics, lozhkin@cs.msu.ru

Abstract — **In this paper, we consider the problem of exact synthesis of inverter graphs for Boolean functions of small number of variables. Main task was to construct catalogue of such schemes. This problem is considered for two models of inverter graphs: and-inverter graph and majority-inverter graph. Algorithms for exact synthesis and corresponding software have been developed to solve this problem. These methods include re-writing, SAT-based synthesis, graph enumeration, iterative minimization and can be applied to any inverter graph model. The qualitative analysis of these methods has been performed. Near-optimal inverter graphs for all functions of five or less variables were obtained. We have all optimal majority-inverter graph realizations with complexity equal or less than 4. An upper bound of the complexity for all functions of five variables was established 11 for the MIG model and 17 for the AIG model. The average complexity of majority-inverter graphs in our database is 7.701 and the average complexity of and-inverter graphs is 12.800. As a result, average complexity for MIG model was shown to be approximately 39.8% less than the average complexity of AIG model. An implementation in the MIG model was demonstrated for the single worst-case equivalence class of Boolean functions.**

Keywords — **methods for exact synthesis, Boolean logic network, Post basis, complexity of Boolean functions, average complexity, majority function, circuit design database.**

References

[1] Luca Amaru´, Pierre-Emmanuel Gaillardon и Giovanni Micheli. «Majority-Inverter Graph: A Novel Data-Structure and Algorithms for Efficient Logic Optimization». // Proceedings of the 51st Design Automation Conference (DAC), 2014. DOI: 10.1145/2593069.2593158.

[2] Per Bjesse и Arne Boralv. «DAG-aware circuit compression for formal verification», 2004. PP. 42—49. ISBN: 0-7803-8702-3. DOI: 0.1109/ICCAD.2004.1382541.

[3] John Darringer, William Jr, C. Berman и Louise Trevillyan. «Logic Synthesis Through Local Transformations». // IBM Journal of Research and Development 25, 1981. PP. 272—280. DOI: 10.1147/rd.254.0272.

[4] Kanupriya Gulati. Advanced Techniques in Logic Synthesis, Optimizations and Applications, 2011. ISBN: 978-1-4419-7517-1. DOI: 10.1007/978-1-4419-7518-8.

[5] Winston Haaswijk, Mathias Soeken, Alan Mishchenko и Giovanni Micheli. «SAT- Based Exact Synthesis: Encodings, Topology Families, and Parallelism». // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2019. DOI: 10.1109/TCAD.2019.2897703.

[6] Leo Hellerman. «A Catalog of Three-Variable Or-Invert and And-Invert Logical Circuits». // Electronic Computers, IEEE Transactions on EC-12, 1963. PP. 198 — 223. DOI: 10.1109/PGEC.1963.263531.

[7] Heinz Riener, Winston Haaswijk, Alan Mishchenko, Giovanni De Micheli и Mathias Soeken. «On-the-fly and DAG-aware: Rewriting Boolean Networks with Exact Synthesis». // 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE), PP. 1649—1654.

[8] L. Scheffer, L. Lavagno и G. Martin. EDA for IC implementation, circuit design, and process technology. 2006, PP. 1—596.

[9] Mathias Soeken, Luca Amaru, Pierre-Emmanuel Gaillardon и Giovanni Micheli. «Exact Synthesis of Majority-Inverter Graphs and Its Applications». // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 2017, DOI: 10.1109/TCAD.2017.2664059.

[10] Mathias Soeken, Luca Gaetano Amaru`, Pierre-Emmanuel Gaillardon и Giovanni De Micheli. «Optimizing Majority-Inverter Graphs with Functional Hashing». // Proceedings of the 2016 Conference on Design, Automation & Test in Europe. DATE '16. Dresden, Germany: EDA Consortium, 2016, PP. 1030–1035. ISBN: 9783981537062.

[11] S.A. Lozhkin, M.S. Shupletsov, V.A. Konovodov, B.R. Danilov, V.V. Zhukov, N.Yu. Bagrov. «Raspredelennaya sistema i algoritmy poiska minimal'nyh i blizkih k nim kontaktnyh skhem dlya bulevyh funkcij ot malogo chisla peremennyh». (Distributed system and switching circuits optimization methods for Boolean functions of small number of variables) // Sbornik trudov Vserossijskih nauchno-tekhnicheskih konferencij "Problemy razrabotki perspektivnyh mikro i nanoelektronnyh sistem (MES)", 1 (2016), PP. 40—47. (in Russian)

[12] S. S. Marchenkov. Zamknutye klassy bulevyh funkcij. (Closed classes of Boolean functions) FIZMATLIT Moskva, 2000. ISBN: 5-9221-0066-1. (in Russian)

[13] G.S. Tseitin. On the complexity of derivation in propositional calculus, 1968. In: Slisenko AO (ed) Studies in constructive mathematics and mathematical logic. Seminars in mathematics, vol 8. Steklov Mathematical Institute, Leningrad, Russia, pp 234–259 (English Translation: Consultants Bureau, New York, 1970, pp 115–125)