УДК 517.98

П.Н. Бибило, Ю.Ю. Ланкевич

МИНИМИЗАЦИЯ МНОГОУРОВНЕВЫХ ПРЕДСТАВЛЕНИЙ СИСТЕМ БУЛЕВЫХ ФУНКЦИЙ НА ОСНОВЕ РАЗЛОЖЕНИЯ ШЕННОНА

Предлагается приближенный алгоритм формирования перестановки переменных, по каждой из которых последовательно строятся разложения Шеннона системы дизъюнктивных нормальных форм полностью определенных булевых функций с целью получения многоуровневого представления функций, называемого в литературе сокращенной упорядоченной диаграммой двоичного выбора либо диаграммой двоичных решений. Приводятся результаты экспериментального сравнения программы, реализующей предложенный алгоритм, с программой, реализующей алгоритм перебора случайных перестановок.

Введение

Разложение Шеннона для булевых функций было предложено давно [1] и позволяет уменьшать число переменных подфункций (коэффициентов разложения). На этом основываются методы синтеза комбинационных схем в различных базисах логических элементов [2]. Получающиеся многоуровневые представления функций и систем реализуются каскадными логическими схемами. Для представления булевых функций и систем таких функций разложения Шеннона исходных функций (и получающихся в процессе разложения подфункций) стали задавать в виде ориентированных графов BDD (Binary Decision Diagram – диаграмма двоичного выбора, диаграмма двоичных решений) [3–7]. По сути, BDD задают ортогонализованные многоуровневые представления (формы) булевых функций, что позволяет эффективно выполнять логические операции над BDD [2, 3, 7]. Широкое распространение получили представления в виде BDD при решении задач верификации проектов цифровых систем [8]. Например, в верификаторе SMV, реализующем метод символьной верификации Model Checking [9], применяется алгоритм верификации моделей, основанный на OBDD (Ordered Binary Decision Diagram – упорядоченная диаграмма двоичного выбора).

Многоуровневые разложения систем функций на базе разложения Шеннона, полученные в результате логической оптимизации, широко используются при синтезе логических схем на этапе технологически независимой оптимизации, например, в промышленном синтезаторе LeonardoSpectrum [10]. В работе [2] показано, что глобальная оптимизация комбинационной логики на основе разложения Шеннона позволяет значительно улучшать результаты синтеза, так как уменьшение сложности (площади) синтезируемых логических схем основано на предварительной технологически независимой оптимизации – нахождении одинаковых коэффициентов в разложениях Шеннона функций реализуемой системы. Основной проблемой при оптимизации таких многоуровневых представлений булевых функций является нахождение последовательности (перестановки) переменных, по которым проводятся разложения.

В настоящей работе предлагается приближенный алгоритм нахождения такой перестановки переменных, которая ориентирована на уменьшение сложности BDD, представляющей систему булевых функций. Предложенный алгоритм программно реализован. Для систем функций большой размерности (25 и более переменных) он оказался более предпочтительным по сравнению с алгоритмом [11] перебора случайно генерируемых перестановок, так как при последующем синтезе позволяет получать схемы меньшей площади и большего быстродействия, что подтверждается проведенным экспериментальным сравнением программ, реализующих данные алгоритмы.

1. Основные определения

Булевыми называются функции $f(\boldsymbol{x}) = f(x_1, x_2, ..., x_n)$, принимающие значения 0, 1 и зависящие от переменных $x_1, x_2, ..., x_n$, которые также принимают значения 0, 1. Широко распространенной формой задания булевых функций являются ДНФ (дизъюнктивные нормальные

формы) – дизъюнкции элементарных конъюнкций. ДНФ обычно представляется троичной матрицей. Например, ДНФ D^1 булевой функции $f^1 = x_2x_3x_4\bar{x}_5\bar{x}_6 \vee \bar{x}_2\bar{x}_3x_5x_6 \vee \bar{x}_1x_2x_4$ может быть представлена троичной матрицей (табл. 1), строки которой соответствуют элементарным конъюнкциям. В данной таблице троичный вектор (0 1 – 1 – –) задает элементарную конъюнкцию $\bar{x}_1x_2x_4$: литералы x_2, x_4 представляются единицами в троичном векторе, литерал $\bar{x}_1$ – нулем, отсутствующие литералы переменных x_3, x_5, x_6 – символом «–». Характеристическое множество M^{f^1} булевой функции, заданной ДНФ из табл. 1, представлено в табл. 2. В характеристическое множество включаются только те наборы значений переменных, на которых булева функция принимает единичное значение [12].

Таблица 1

ДНФ D^1 функции f^1

D^1	k_i
x_1 x_2 x_3 x_4 x_5 x_6	
– 1 1 1 0 0	k_1
– 0 0 – 1 1	k_2
0 1 – 1 – –	k_3

Таблица 2

Характеристическое множество M^{f^1} функции f^1

M^{f^1}
x_1 x_2 x_3 x_4 x_5 x_6
0 1 1 1 0 0
1 1 1 1 0 0
0 0 0 0 1 1
0 0 0 1 1 1
1 0 0 0 1 1
1 0 0 1 1 1
0 1 0 1 0 0
0 1 0 1 0 1
0 1 0 1 1 0
0 1 0 1 1 1
0 1 1 1 0 1
0 1 1 1 1 0
0 1 1 1 1 1

Если исходные булевы функции представлены в ДНФ, то основной задачей при оптимизации их многоуровневых представлений на базе разложения Шеннона является сравнение ДНФ на равносильность. Обозначим через D ДНФ, которая представляет булеву функцию $f = f(x_1,...,x_n)$ с характеристическим множеством M^f. Пусть M^k – характеристическое множество элементарной конъюнкции k. Будем говорить, что элементарная конъюнкция k имплицирует ДНФ D тогда и только тогда, когда $M^k \subseteq M^f$, т. е. когда характеристическое множество M^k конъюнкции включается в характеристическое множество той булевой функции, которую представляет ДНФ D. Например, элементарная конъюнкция $k_1 = \bar{x}_1\bar{x}_2\bar{x}_3x_4\bar{x}_5$ с характеристическим множеством M^{k_1}={010100, 010101}, представленная троичным вектором (0 1 0 1 0 –), имплицирует D^1 (табл. 1). Элементарная конъюнкция $k_2 = x_1x_2x_3x_4\bar{x}_5$ с характеристическим множеством M^{k_2}={111100, 111101}, представленная троичным вектором (1 1 1 1 0 –), не имплицирует D^1, так как набор 111101 не принадлежит M^{f^1} и поэтому условие $M^{k_2} \subseteq M^{f^1}$ не выполняется.

Рассмотрим ДНФ D^h булевой функции $h = h(x_1,...,x_n)$, имеющей характеристическое множество M^h, и ДНФ D^g булевой функции $g = g(x_1,...,x_n)$ с характеристическим множеством M^g. Очевидно, что ДНФ D^h, D^g являются равносильными (записывается $D^h = D^g$) тогда и только тогда, когда $M^h = M^g$.

В литературе хорошо известно, что использование для сравнения ДНФ функций построения их характеристических множеств является неэффективным, когда число переменных функций превышает три десятка [12, 13]. Для сравнения ДНФ будем применять способ, основанный на операциях над ДНФ и следующем утверждении [12]: ДНФ $D^h = k_1^h \vee k_2^h \vee ... \vee k_{p_1}^h$ и $D^g = k_1^g \vee k_2^g \vee ... \vee k_{p_2}^g$ являются равносильными тогда и только тогда, когда каждая из конъюнкций k_i^h ДНФ D^h имплицирует ДНФ D^g (это записывается в виде $(k_i^h \to D^g) = 1$), а каждая из конъюнкций k_i^g ДНФ D^g имплицирует ДНФ D^h (записывается в виде $(k_i^g \to D^h) = 1$). Далее будет приведен алгоритм сравнения двух ДНФ, базирующийся на данном утверждении.

Под векторной булевой функцией $\boldsymbol{f}(\boldsymbol{x})$, $\boldsymbol{x} = (x_1, x_2, ..., x_n)$, будем понимать упорядоченную систему булевых функций $\boldsymbol{f}(\boldsymbol{x}) = (f^1(\boldsymbol{x}), \ldots, f^m(\boldsymbol{x}))$. Широко известной в литературе формой представления систем булевых функций является пара матриц $<T^x, B^f>$: троичная матрица T^x задания элементарных конъюнкций в виде троичных векторов и булева матрица B^f вхождений конъюнкций в ДНФ компонентных функций системы. В табл. 3 дан пример задания системы ДНФ векторной полностью определенной функции $\boldsymbol{f}(\boldsymbol{x}) = (f^1(\boldsymbol{x}), f^2(\boldsymbol{x}), f^3(\boldsymbol{x}))$, где $f^1 = x_2 x_3 x_4 \bar{x}_5 \bar{x}_6 \vee \bar{x}_2 x_3 x_5 x_6 \vee \bar{x}_1 x_2 x_4$; $f^2 = x_2 \bar{x}_3 \bar{x}_5 \bar{x}_6 \vee \bar{x}_1 \bar{x}_2 \bar{x}_3 \vee x_1 x_2 x_3 x_4 \bar{x}_6 \vee \bar{x}_2 x_3 \bar{x}_4 x_5 x_6$; $f^3 = \bar{x}_1 x_3 \bar{x}_4 x_6 \vee x_3 \bar{x}_4 x_5 x_6 \vee \bar{x}_1 x_3 x_5 x_6 \vee x_2 \bar{x}_3 x_4 \bar{x}_5 \bar{x}_6 \vee \bar{x}_2 x_3 \bar{x}_4 \vee \bar{x}_1 x_2 x_4$. Здесь троичный вектор (0 1 – 1 – –) (последняя строка матрицы T^x) задает элементарную конъюнкцию $\bar{x}_1 x_2 x_4$, входящую в ДНФ компонентных функций f^1 и f^3.

Таблица 3

Пример системы ДНФ трех булевых функций

T^x	B^f
$x_1\ x_2\ x_3\ x_4\ x_5\ x_6$	$f^1\ f^2\ f^3$
– 1 1 1 0 0	1 0 0
– 0 0 – 1 1	1 0 0
0 1 – 1 – –	1 0 1
– 1 0 – 0 0	0 1 0
0 1 0 – – –	0 1 0
1 1 1 1 – 0	0 1 0
– 0 1 0 1 1	0 1 0
0 – 1 0 – 1	0 0 1
– – 1 0 1 1	0 0 1
0 – 1 – 1 1	0 0 1
– 1 0 1 0 0	0 0 1
– 1 1 0 – –	0 0 1

Второй используемой в данной работе формой задания систем булевых функций являются алгебраические многоуровневые представления на базе разложения Шеннона.

Разложением Шеннона полностью определенной булевой функции $f = f(x)$ по переменной x_i называется представление

$$f = f(\boldsymbol{x}) = \bar{x}_i f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \vee x_i f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n). \quad (1)$$

Функции $f_0 = f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n)$, $f_1 = f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n)$ в правой части (1) называются остаточными функциями либо коэффициентами разложения. Они полу-

чаются из функции $f = f(x_1,...,x_n)$ подстановкой вместо переменной x_i констант 0 и 1 соответственно. Каждый из коэффициентов $f_0 = f(x_1,...,x_{i-1},\ 0,\ x_{i+1},...,\ x_n)$ и $f_1 = f(x_1,...,x_{i-1},\ 1,\ x_{i+1},...,\ x_n)$ может быть разложен по одной из переменных из множества $\{x_1,...,x_{i-1},x_{i+1},...,\ x_n\}$. Процесс разложения коэффициентов заканчивается, когда все n переменных будут использованы для разложения, либо когда все коэффициенты выродятся до констант 0, 1. На каждом шаге разложения выполняется сравнение на равенство полученных коэффициентов и оставляется один из нескольких попарно равных коэффициентов. Если же коэффициенты разложения по переменной x_i равны

$$f(x_1,...,x_{i-1},\ 0,\ x_{i+1},...,\ x_n) = f(x_1,...,x_{i-1},\ 1,\ x_{i+1},...,\ x_n),$$

то переменная x_i называется несущественной (фиктивной) и $f = f(x_1,...,x_{i-1},\ x_{i+1},...,\ x_n)$.

Под BDD понимается ориентированный ациклический граф, задающий последовательные разложения Шеннона булевой функции $f(x_1,...,x_n)$ по всем ее переменным $x_1,...,x_n$ при заданном порядке (перестановке) переменных, по которым проводятся разложения. BDD одной полностью определенной булевой функции содержит три вида вершин [2]: функциональные вершины, соответствующие разлагаемым функциям либо коэффициентам; вершины-переменные; листовые вершины, соответствующие константам 0, 1. Функциональная вершина, соответствующая функции f, называется корнем. Если коэффициенты разложения (1) равны, то граф BDD упрощается, так как вершина x_i, из которой исходит одна дуга, удаляется из графа. BDD, представляющая систему *m* полностью определенных булевых функций, имеет *m* корневых и две листовые вершины 0, 1, которые обычно дублируются для упрощения изображения графа. Ориентация дуг не показывается, так как при изображении BDD все дуги ориентируются сверху вниз. Под сложностью S_{BDD} понимается число функциональных вершин BDD [2].

Заметим, что представление многоуровневого разложения системы булевых функций в виде графа в данной статье используется для иллюстрации зависимостей булевых переменных в логических выражениях. Целью является получение минимального числа логических выражений, по которым впоследствии осуществляется синтез логической схемы в заданном базисе (библиотеке) логических элементов. Каждой функциональной вершине BDD соответствует логическое выражение, представляющее формулу (1) разложения Шеннона либо редуцированную (сокращенную) форму в тех случаях, когда один из коэффициентов равен 0 либо 1. Поэтому функциональные вершины нижнего (неконстантного) уровня BDD равны переменной либо ее инверсии. В виде уравнений такие функции не будут записываться, поэтому число уравнений многоуровневого представления обычно меньше сложности BDD на одну либо две функциональные вершины.

2. Постановка задачи

BDD задает в виде графа последовательность разложений Шеннона исходной функции и получаемых коэффициентов разложения по некоторой перестановке переменных. Минимизация сложности BDD основана на том, что в процессе разложения могут появляться одинаковые коэффициенты разложения не только у одной, но и у нескольких (либо даже у всех) компонентных функций. Давно известно, что от перестановки переменных, по которым ведется разложение, зависит сложность (число вершин) BDD, поэтому основной задачей компактного представления булевой функции либо системы функций в виде BDD является нахождение перестановки, дающей минимальное число вершин BDD.

Под BDD-минимизацией в данной работе будет пониматься оптимизация многоуровневых представлений систем булевых функций, соответствующих сокращенным упорядоченным BDD (ROBDD, от англ. Reduced Ordered BDD). Подробное описание упорядоченных BDD дано в [5], ROBDD – в [7]. Далее под BDD будут всегда пониматься ROBDD.

Задача. Для заданной системы ДНФ булевых функций найти такую перестановку $< x_{i_1}, x_{i_2}, ..., x_{i_n} >$ переменных, по которой будет получена BDD минимальной сложности.

Сложность BDD зависит от порядка следования (перестановки) переменных разложения. Так как число всех перестановок переменных для булевой функции $f = f(x_1, x_2, ..., x_n)$, зависящей от n переменных, равно $n!$ (факториалу числа n), то нахождение лучшей перестановки, т. е. перестановки, обеспечивающей минимальную сложность BDD для больши́х (сотни и более) значений числа n, представляет собой такую трудоемкую задачу, что точное ее решение в настоящее время невозможно получить. Поэтому основными методами нахождения лучшей перестановки, предложенными в литературе [14–18], являются методы генерирования случайных перестановок, дополненные различными эвристиками. Полный перебор всех $n!$ перестановок осуществим для небольших значений n. Например, в работе [11] были испытаны все перестановки переменных для примеров систем ДНФ со значениями $n \le 8$.

3. Представление одной ДНФ и систем ДНФ

При программной реализации операций над конъюнкциями и ДНФ было использовано представление элементарных конъюнкций в виде $2n$ разрядных булевых векторов, при этом инверсному литералу $\bar{x}_i$ присваивался код 01, литералу x_i – код 10, отсутствующий в элементарной конъюнкции литерал получал код 11, кодовая комбинация 00 при таком кодировании считается недопустимой и ее наличие является ошибкой. Например, элементарные конъюнкции ДНФ (см. табл. 1) представляются следующими булевыми векторами:

111010100101 (k_1) (3749)
110101**11**1010 (k_2) (3450)
0110**11**10**1111** (k_3) (1775).

Жирным шрифтом выделены кодовые комбинации 11, соответствующие отсутствующим литералам. Булевы векторы, представляющие элементарные конъюнкции, можно интерпретировать как целые положительные числа и упорядочить по возрастанию (неубыванию) их десятичного эквивалента. Каждая из ДНФ системы представлялась в подобном виде, после чего строилась упорядоченная последовательность всех ДНФ системы булевых функций. Если элемент (число) из задания ДНФ D^i был больше (не меньше) соответствующего числа из задания ДНФ D^j, то D^i располагалась раньше ДНФ D^j в общей упорядоченной последовательности их задания.

4. Алгоритм нахождения перестановки переменных

Предлагаемый эвристический алгоритм решения задачи является итерационным и выполняется до тех пор, пока текущая система функций (коэффициентов разложений) не выродится до констант 0, 1. На первой итерации в качестве текущей берется исходная система функций. На каждой итерации j (j = 1, …, n) выполняются следующие шаги:

Шаг 1. Выбор переменной x_j, по которой производится разложение Шеннона всех функций текущей системы.

1.1. Построение разложений Шеннона по каждой из переменных $x_1, x_2, ..., x_n$ каждой из ДНФ текущей системы функций системы, т. е. получение ДНФ коэффициентов, не равных нулю. Исключение из рассмотрения коэффициентов, равных нулю.

1.2. Отдельная минимизация в классе ДНФ каждого из полученных коэффициентов. Для этого итеративным образом выполняются всевозможные склеивания и поглощения элементарных конъюнкций, входящих в минимизируемую ДНФ. Конъюнкции исходной ДНФ разбиваются на такие подмножества, что все конъюнкции в одном подмножестве содержат одинаковое число единиц. Проверка на склеивание конъюнкций осуществляется между векторами

соседних подмножеств, т. е. векторами, содержащими k и k +1 единицу, точно так же, как это выполняется в методе Квайна – Мак-Класки [12]. После выполнения всевозможных склеиваний выполняются поглощения и начинается следующая итерация склеиваний и поглощений. Итерации заканчиваются тогда, когда нет склеивающихся пар конъюнкций. Таким образом, минимизация осуществляется с помощью приближенного алгоритма.

1.3. Проверка полученных минимизированных ДНФ на равносильность, в результате чего из всего множества ДНФ, представляющих коэффициенты, формируется множество M_i^j попарно различных ДНФ.

1.4. Исключение из дальнейшего рассмотрения коэффициентов, равных константе 1.

1.5. Оценка переменных, от которых зависят функции текущей системы. Каждая переменная x_i текущей системы оценивается числом S_i^j – мощностью $\left|M_i^j\right| = S_i^j$ множества M_i^j различных коэффициентов, полученных при разложении всех функций текущей системы по переменной x_i.

1.6. В качестве переменной x_j для разложения Шеннона на итерации j выбирается переменная x_i, оцениваемая минимальным числом S_i^j. Если таких переменных несколько, то из них выбирается первая по порядку. Переход на шаг 2.

Шаг 2. Построение разложений Шеннона текущей системы функций и формирование системы функций для итерации j+1.

2.1. Построение разложений Шеннона функций текущей системы (на итерации j) по выбранной переменной x_i, т. е. получение ДНФ коэффициентов, не равных 0. Исключение из рассмотрения коэффициентов, равных 0.

2.2. Минимизация ДНФ, представляющих коэффициенты, так же, как она выполнялась на шаге 1.2.

2.3. Проверка полученных минимизированных ДНФ на равносильность и формирование множества M_i^j попарно различных ДНФ из всего множества ДНФ, представляющих коэффициенты. Сравнению на равносильность подвергались пары тех ДНФ, которые были соседними в общей упорядоченной последовательности их задания. Алгоритм проверки ДНФ на равносильность описывается далее.

2.4. Исключение из множества M_i^j (т. е. из дальнейшего рассмотрения) коэффициентов, равных 1.

2.5. Проверка, является ли множество M_i^j пустым, т. е. остались ли в множестве M_i^j такие коэффициенты, которые не равны константе 1. Если M_i^j оказалось пустым, то переход на шаг 3. В противном случае множество минимизированных ДНФ множества M_i^j является текущей системой функций для итерации j+1.

Шаг 3. Конец.

Описанный выше алгоритм позволяет найти перестановку переменных для проведения разложений Шеннона по всем переменным. После того как перестановка сформирована и BDD построена, выполняется процедура сокращения BDD. Выполнение процедуры сокращения BDD требуется потому, что алгоритм сравнения коэффициентов на равенство является приближенным и не все пары ДНФ сравнивались на равносильность. Описание процедуры сокращения BDD дано в [4, 7]. Она основывается на том, что функции (коэффициенты) равны тогда и только тогда, когда равны их BDD, при этом предполагается один и тот же порядок переменных для разложения. Алгоритм R [7, с. 257] сокращения (приведения) BDD формулируется в терминах задания графов BDD в виде списков.

Если число выполненных итераций алгоритма оказалось меньше числа n, то это означает, что некоторые переменные оказались несущественными для каждой из функций исходной системы.

5. Алгоритм сравнения ДНФ

Алгоритм сравнения ДНФ $D^h = k_1^h \vee k_2^h \vee ... \vee k_{p_1}^h$, $D^g = k_1^g \vee k_2^g \vee ... \vee k_{p_2}^g$ заключается в проверке равенств

$$(k_i^h \to D^g) = 1; \tag{2}$$

$$(k_j^g \to D^h) = 1, \tag{3}$$

где i=1, 2,…, p_1, j=1, 2,…, p_2, и является приближенным. Он точно гарантирует установление неравносильности ДНФ, однако некоторые равносильные ДНФ может посчитать неравносильными. Именно это обстоятельство и требует выполнения после алгоритма нахождения перестановки процедуры сокращения BDD.

Шаг 1. Нахождение тождественно равных конъюнкций, входящих как в ДНФ D^h, так и в ДНФ D^g, и исключение этих конъюнкций из проверок (2), (3).

Шаг 2. Проверка выполнения условий (2), (3) для оставшихся конъюнкций. Если все равенства (2), (3) выполняются, то ДНФ равносильны: $D^h = D^g$. Если же хотя бы одно из равенств (2), (3) не выполняется, то ДНФ неравносильны: $D^h \neq D^g$.

Проверка равенства $(k_i^h \to D^g)$=1 осуществляется следующим образом: вычисляется ДНФ $D_i^{h,g} = k_i^h \,\&\, D^g$, затем ДНФ $D_i^{h,g}$ минимизируется в классе ДНФ с помощью алгоритма, описанного на шаге 1.2 алгоритма нахождения перестановки переменных. Если в результате минимизации будет получена элементарная конъюнкция k_i^h, то считается, что $(k_i^h \to D^g) = 1$. Поскольку алгоритм минимизации ДНФ является приближенным, то это означает, что ДНФ $D_i^{h,g}$ (возможно, равная k_i^h) может оказаться представленной несколькими элементарными конъюнкциями (а не одной элементарной конъюнкцией k_i^h). Такое представление приведет к выводу о том, что равенство $(k_i^h \to D^g) = 1$ не выполняется, и будет считаться, что $D^h \neq D^g$, хотя на самом деле данные ДНФ равносильны.

Проведем сравнение на равносильность ДНФ D^h (табл. 4) и D^g (табл. 5). Так как $k_1^h = k_1^g$, $k_2^h = k_2^g$, $k_4^h = k_3^g$, то осталось проверить выполнение равенства $(k_3^h \to D^g)$=1. Вычислим ДНФ $D_3^{h,g}$: $D_3^{h,g} = k_3^h \,\&\, D^g = \bar{x}_1 x_3 x_4 (\bar{x}_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 \bar{x}_3 \bar{x}_4 \vee x_2 x_3 x_4) = \bar{x}_1 \bar{x}_2 x_3 x_4 \vee \bar{x}_1 x_2 x_3 x_4 =$ $= \bar{x}_1 x_3 x_4 = k_3^h$. Таким образом, $(k_3^h \to D^g)$=1 и $D^h = D^g$.

Таблица 4

ДНФ функции h

D^h	k_i^h
$x_1\ x_2\ x_3\ x_4$	
0 0 1 –	k_1^h
0 1 0 0	k_2^h
0 – 1 1	k_3^h
– 1 1 1	k_4^h

Таблица 5

ДНФ функции g

D^g	k_i^g
$x_1\ x_2\ x_3\ x_4$	
0 0 1 –	k_1^g
0 1 0 0	k_2^g
– 1 1 1	k_3^g

Таблица 6

ДНФ функции r

D^r	k_i^r
$x_1\ x_2\ x_3\ x_4$	
0 0 1 –	k_1^r
0 1 0 0	k_2^r
0 – 0 1	k_3^r
– 1 1 1	k_4^r

Проведем сравнение на равносильность ДНФ D^h (табл. 4) и D^r (табл. 6). В этом случае требуется проверить два равенства: $(k_3^h \to D^r)$=1 и $(k_3^r \to D^h)$=1. Для проверки выполнения $(k_3^h \to D^r)$=1 вычислим $D_3^{h,r} = k_3^h \,\&\, D^r = \bar{x}_1 x_3 x_4 (\bar{x}_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 \bar{x}_3 \bar{x}_4 \vee \bar{x}_1 \bar{x}_3 x_4 \vee x_2 x_3 x_4) =$

$= \bar{x}_1\bar{x}_2x_3x_4 \vee \bar{x}_1x_2x_3x_4 = \bar{x}_1x_3x_4 = k_3^h$, т. е. равенство $(k_3^h \to D^r)$=1 выполняется. Для проверки выполнения равенства $(k_3^r \to D^h)$=1 вычислим $D_3^{r,h} = k_3^r \,\&\, D^h = \bar{x}_1\bar{x}_3x_4(\bar{x}_1\bar{x}_2x_3 \vee \bar{x}_1x_2\bar{x}_3\bar{x}_4 \vee \vee \bar{x}_1x_3x_4 \vee x_2x_3x_4) = 0 \neq k_3^r$. Очевидно, что равенство $(k_3^r \to D^h)$=1 не выполняется, это означает $D^h \neq D^r$.

6. Пример

Проиллюстрируем алгоритм нахождения перестановки переменных на примере векторной функции $f(\boldsymbol{x})$, состоящей из трех компонентных функций f^1, f^2, f^3 (см. табл. 3). Как будет показано далее, лучшей перестановкой оказывается $< x_3, x_4, x_1, x_5, x_2, x_6 >$.

Формулы разложения Шеннона по переменной x_3 записываются в виде

$$f^1(\boldsymbol{x}) = \bar{x}_3 h^1(x_1, x_2, x_4, x_5, x_6) \vee x_3 h^2(x_1, x_2, x_4, x_5, x_6),$$

$$f^2(\boldsymbol{x}) = \bar{x}_3 h^6(x_1, x_2, x_4, x_5, x_6) \vee x_3 h^3(x_1, x_2, x_4, x_5, x_6),$$

$$f^3(\boldsymbol{x}) = \bar{x}_3 h^2(x_1, x_2, x_4, x_5, x_6) \vee x_3 h^4(x_1, x_2, x_4, x_5, x_6).$$

Коэффициенты, входящие в данные формулы, представлены в табл. 7.

Таблица 7

Коэффициенты разложения системы ДНФ по переменной x_3

Коэффициенты	ДНФ
	x_1 x_2 x_4 x_5 x_6
$h^1 = h^1(x_1, x_2, x_4, x_5, x_6) = f^1(x_1, x_2, 0, x_4, x_5, x_6)$	– 0 – 1 1 0 1 1 – –
$h^2 = h^2(x_1, x_2, x_4, x_5, x_6) = f^1(x_1, x_2, 1, x_4, x_5, x_6) = f^3(x_1, x_2, 0, x_4, x_5, x_6)$	– 1 1 0 0 0 1 1 – –
$h^3 = h^3(x_1, x_2, x_4, x_5, x_6) = f^2(x_1, x_2, 1, x_4, x_5, x_6)$	1 1 1 – 0 – 0 0 1 1
$h^4 = h^4(x_1, x_2, x_4, x_5, x_6) = f^3(x_1, x_2, 1, x_4, x_5, x_6)$	0 – 0 – 1 – – 0 1 1 0 – – 1 1 – 1 0 – – 0 1 1 – –
$h^6 = h^6(x_1, x_2, x_5, x_6) = f^2(x_1, x_2, 0, x_4, x_5, x_6)$	– 1 – 0 0 0 1 – – –

Очевидно, что коэффициент $h^6 = x_2\bar{x}_5\bar{x}_6 \vee \bar{x}_1x_2$ не зависит от переменной x_4. Заметим, что приближенный алгоритм минимизации ДНФ на первом шаге разложения не позволяет уменьшить число элементарных конъюнкций в коэффициентах (табл. 7). На первой итерации алгоритма поиска перестановки (табл. 8) переменная x_3 имеет наименьшее значение S_3^1=5, поэтому первой переменной для разложения системы функций выбирается x_3.

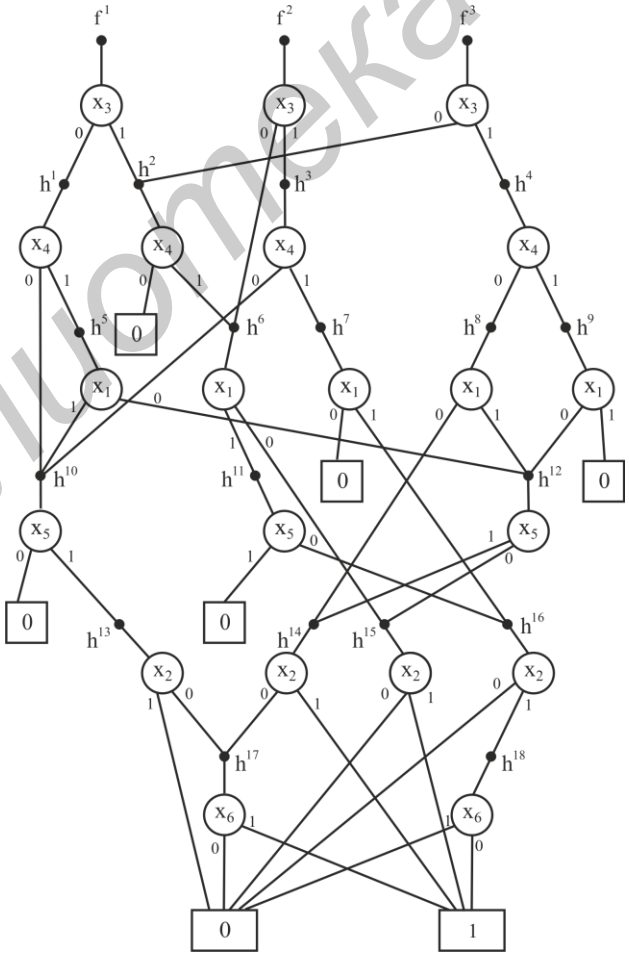
Результаты оценки переменных и их выбор по минимальному значению S_i^j приведены в табл. 8.

Таблица 8

Выбор переменных для разложения Шеннона системы ДНФ функций

i	x_i	Итерация алгоритма нахождения перестановки переменных											
		Первая		Вторая		Третья		Четвертая		Пятая		Шестая	
		S_i^1	x_3	S_i^2	x_4	S_i^3	x_1	S_i^4	x_5	S_i^5	x_2	S_i^6	x_6
1	x_1	6		10		6	min						
2	x_2	6		8		6		4		2	min		
3	x_3	5	min										
4	x_4	6		6	min								
5	x_5	6		9		8		4	min				
6	x_6	6		9		8		4		2		0	min

На четвертой итерации три переменные x_2, x_5, x_6 имеют одинаковые веса $S_2^4 = S_5^4 = S_6^4 = 4$ и для очередного разложения выбрана переменная x_5 для того, чтобы проиллюстрировать возможность дополнительного перебора в таких ситуациях. Как было сказано выше, результирующая перестановка для выполнения последовательных разложений Шеннона системы ДНФ функций (см. табл. 3) имеет вид $< x_3, x_4, x_1, x_5, x_2, x_6 >$. Полученный граф BDD (рисунок) содержит 18 функциональных вершин, поэтому $S_{BDD} = 18$.



BDD, задающая многоуровневое представление системы ДНФ

Формулы многоуровневого представления системы функций по полученной перестановке $< x_3, x_4, x_1, x_5, x_2, x_6 >$ легко записать по BDD:

$$f^1 = \bar{x}_3 h^1 \vee x_3 h^2,\ f^2 = \bar{x}_3 h^6 \vee x_3 h^3,\ f^3 = \bar{x}_3 h^2 \vee x_3 h^4,$$

$$h^1 = \bar{x}_4 h^{10} \vee x_4 h^5,\ h^2 = x_4 h^6,\ h^3 = \bar{x}_4 h^{10} \vee x_4 h^7,\ h^4 = \bar{x}_4 h^8 \vee x_4 h^9,$$

$$h^5 = \bar{x}_1 h^{12} \vee x_1 h^{10},\ h^6 = \bar{x}_1 h^{15} \vee x_1 h^{11},\ h^7 = x_1 h^{16},\ h^8 = \bar{x}_1 h^{14} \vee x_1 h^{12},\ h^9 = \bar{x}_1 h^{12},$$

$$h^{10} = x_5 h^{13},\ h^{11} = \bar{x}_5 h^{16},\ h^{12} = \bar{x}_5 h^{15} \vee x_5 h^{14},$$

$$h^{13} = \bar{x}_2 h^{17},\ h^{14} = \bar{x}_2 h^{17} \vee x_2,\ h^{15} = x_2;\ h^{16} = x_2 h^{18},$$

$$h^{17} = x_6,\ h^{18} = \bar{x}_6.$$

Каждой из 18 функциональных вершин BDD соответствует полная либо редуцированная формула разложения Шеннона.

7. Результаты эксперимента

Предложенный в данной статье алгоритм поиска лучшей перестановки был программно реализован, разработанная программа *BDD_Builder* сравнивалась с программой *Tie_BDD*, выполняющей алгоритм [11] случайного перебора перестановок. Сравнение проводилось на потоке примеров систем ДНФ векторных функций $\boldsymbol{f}(\boldsymbol{x}) = (f^1(\boldsymbol{x}), \ldots, f^m(\boldsymbol{x}))$, $\boldsymbol{x} = (x_1, \ldots, x_n)$, из библиотеки *Berkeley PLA Test Set* [19] примеров систем ДНФ и трех примеров схем (X3, Frg2, Too_large) многоуровневой логики, преобразованных в системы ДНФ. Пример Sin_16 – это задание в виде таблицы истинности 16-битного приближения тригонометрической функции $y = \sin x$; примеры Syst4, Syst8 взяты из практики промышленного проектирования цифровых схем.

В табл. 9 представлены результаты экспериментального сравнения программ *Tie_BDD* и *BDD_Builder*, лучшие решения выделены жирным шрифтом. Обе программы по матричной форме исходной системы ДНФ, представленной на языке SF [20] в формате SDF, получали минимизированное многоуровневое представление системы функций в формате LOG логических выражений. После получения такого представления осуществлялся его перевод в VHDL-описание [21], использующее операторы назначения сигнала и логические операторы AND (И), OR (ИЛИ), NOT (НЕТ). Описание исходной системы ДНФ также переводилось в VHDL-описание для того, чтобы провести с помощью верификатора Formal Pro [22] формальную верификацию исходного VHDL-описания и полученного многоуровневого VHDL-описания. Затем по каждому из VHDL-описаний выполнялось построение логических схем в библиотеке [2] элементов заказной сверхбольшой интегральной схемы, выполнялся подсчет площадей и задержек схем. В качестве синтезатора логических схем из библиотечных элементов использовался Leonardo-Spectrum [10], исходными данными для которого явились соответствующие VHDL-описания логических выражений. Сложность схемы из библиотечных логических элементов на этапе логического проектирования выражалась суммой площадей элементов, составляющих схему. В экспериментах под сложностью многоуровневого представления понималось число R уравнений (при этом уравнения для разложения по последней переменной x_i не записываются, т. е. из записи многоуровневого представления исключаются уравнения вида $h = x_i$, $h = \bar{x}_i$).

В табл. 9 используются следующие обозначения:

n – число переменных $x_1, x_2, \ldots, x_n$;

m – число функций;

k – число различных элементарных конъюнкций, входящих в ДНФ компонентных функций f^j, $j = 1, \ldots, m$;

P – число случайных перестановок, испробованных программой *Tie_BDD*;

R – число логических уравнений многоуровневого разложения Шеннона;

S_{ASIC} – площадь логической схемы, задаваемая в числе базовых ячеек, из которых состоят элементы;

τ – задержка схемы, нс.

Таблица 9

Сравнение программ оптимизации многоуровневых представлений

Имя схемы	n	m	k	Программа *Tie_BDD*				Программа *BDD_Builder*		
				P	R	S_{ASIC}	τ, нс	R	S_{ASIC}	τ, нс
Z5XP1	7	10	128	5000	**65**	**26 499**	**3,85**	**65**	**26 499**	**3,85**
RADD	8	5	120	5000	**27**	**8465**	4,44	33	10 697	**3,85**
ROOT	8	5	256	5000	**73**	**26 717**	**4,81**	**73**	**26 717**	**4,81**
DIST	8	5	256	5000	**144**	**68 601**	5,51	**144**	70 224	**5,38**
LIFE	9	1	512	5000	**24**	14 977	4,99	39	**14 391**	**4,54**
ADDM4	9	8	512	5000	**187**	89 269	7,00	189	**86 959**	**6,25**
Z9SYM	9	1	420	5000	**31**	**18 018**	5,39	**31**	26 499	**3,85**
ADD6	12	7	1092	5000	58	20 222	**5,50**	**54**	**14 419**	5,97
TIAL	14	8	640	5000	**706**	**295 952**	9,56	718	310 148	**8,52**
MP2D	14	14	123	5000	**69**	17 968	4,68	80	**17 438**	**3,56**
IN0	15	11	138	5000	**301**	98 911	7,52	332	**96 389**	**6,70**
Sin_16	16	16	65 536	10	**17 758**	9 822 718	15,86	**17 758**	**9 771 160**	**15,34**
Syst4	17	12	370	500	**307**	134 121	8,56	363	**123 424**	**7,01**
Syst8	25	28	45 548	100	23 821	13 031 482	19,72	**17 437**	**8 183 862**	**19,39**
X9dn_matr	27	7	120	500	204	26 399	5,58	**102**	**23 983**	**5,55**
Vtx1_matr	27	6	110	500	182	28 715	**5,28**	**100**	**25 545**	6,22
Too_large_matr	38	3	1027	500	3519	1 437 218	19,40	**1344**	**488 585**	**14,96**
Soar_matr	83	94	529	500	947	225 934	6,31	**527**	**138 585**	**5,48**
X3_matr	135	99	915	100	1611	338 042	10,69	**855**	**269 151**	**8,76**
Frg2_matr	143	139	3090	100	11 541	2 986 885	17,49	**1532**	**530 785**	**12,56**

Результаты эксперимента позволяют сделать следующие выводы:

1. Оценка сложности системы функций, выражаемая в числе уравнений (либо числе функциональных вершин BDD), является полезной при синтезе схем, так как синтез схемы от представления, в котором меньше уравнений, позволял, как правило, получать схемы меньшей площади. Здесь следует иметь в виду тот факт, что синтезатор LeonardoSpectrum выполняет при синтезе собственные встроенные процедуры технологически независимой оптимизации.

2. Для примеров Z9SYM и ROOT программа *BDD_Builder* нашла BDD минимальной сложности [14]. Результаты решения этих примеров свидетельствуют о том, что локально-оптимальный алгоритм, так же, как и алгоритмы случайного поиска, может найти глобальный оптимум.

3. Предложенный алгоритм выбора перестановки переменных позволяет для практических примеров систем ДНФ (более 25 переменных) получать BDD меньшей сложности, что приводит к логическим схемам меньшей сложности и большего быстродействия. Для систем

ДНФ большой размерности он является более предпочтительным по сравнению с алгоритмом [11] перебора случайных перестановок.

Заключение

Возрастание размерностей задач логического синтеза приводит к необходимости совершенствования методов, алгоритмов и программ технологически независимой оптимизации, выполняемой перед реализацией оптимизированных представлений в требуемый базис логических элементов. Предложенный в данной работе и программно реализованный, экспериментально исследованный алгоритм позволяет эффективно выполнять оптимизацию многоуровневых представлений систем булевых функций с сотнями переменных и функций. Реализующая алгоритм программа может служить эффективным инструментом в системах автоматизированного проектирования для минимизации площади и увеличения быстродействия функциональных блоков заказных цифровых сверхбольших интегральных схем.

Список литературы

1. Шеннон, К. Синтез двухполюсных переключательных схем / К. Шеннон // Работы по теории информации и кибернетике. – М. : ИЛ, 1963. – С. 59–105.

2. Бибило, П.Н. Применение диаграмм двоичного выбора при синтезе логических схем / П.Н. Бибило. – Минск : Беларус. навука, 2014. – 231 с.

3. Akers, S.B. Binary Decision Diagrams / S.B. Akers // IEEE Trans. on Computers. – 1978. – Vol. C–27, no. 6. – P. 509–516.

4. Bryant, R.E. Graph-based algorithms for Boolean function manipulation / R.E. Bryant // IEEE Transactions on Computers. – 1986. – Vol. 35, no. 8. – P. 677–691.

5. Bryant, R.E. Ordered Binary Decision Diagrams / R.E. Bryant, C. Meinel // Logic synthesis and verification. – Boston, Dordrecht, London : Kluwer Academic Publishers, 2002. – P. 285–307.

6. Meinel, C. Algorithms and Data Structures in VLSI Design: OBDD – Foundations and Applications / C. Meinel, T. Theobald. – Berlin, Heidelberg : Springer-Verlag, 1998. – 267 p.

7. Кнут, Д.Э. Искусство программирования / Д.Э. Кнут. – М. : Вильямс, 2013. – Т. 4, А : Комбинаторные алгоритмы, ч. 1. – 960 с.

8. Валидация на системном уровне. Высокоуровневое моделирование и управление тестированием / М. Чэнь [и др.]. – М. : Техносфера, 2014. – 296 с.

9. Карпов, Ю.Г. MODEL CHECKING. Верификация параллельных и распределенных программных систем / Ю.Г. Карпов. – СПб. : БХВ-Петербург, 2010. – 560 с.

10. Бибило, П.Н. Системы проектирования интегральных схем на основе языка VHDL. StateCAD, ModelSim, LeonardoSpectrum / П.Н. Бибило. – М. : СОЛОН-Пресс, 2005. – 384 с.

11. Бибило, П.Н. Алгоритм построения диаграммы двоичного выбора для системы полностью определенных булевых функций / П.Н. Бибило, П.В. Леончик // Управляющие системы и машины. – 2009. – № 6. – С. 42–49.

12. Закревский, А.Д. Логические основы проектирования дискретных устройств / А.Д. Закревский, Ю.В. Поттосин, Л.Д. Черемисинова. – М. : Физматлит, 2007. – 589 с.

13. Закревский, А.Д. Вычисления в многомерном булевом пространстве / А.Д. Закревский. – Минск : ОИПИ НАН Беларуси, 2011. – 106 с.

14. Ishiura, N. Minimization of binary decision diagrams based on exchange of variables / N. Ishiura, H. Sawada, S. Yajima // Computer-Aided Design : proc. 29th IEEE Intern. conf., Santa Clara, 11–14 Nov. 1991 / IEEE Computer Society. – Washington, 1991. – P. 472–475.

15. Jeong, S.-W. An efficient method for optimal BDD ordering computation / S.-W. Jeong, T.-S. Kim, F. Somenzi // VLSI and CAD : proc. of Intern. conf. – Taejon, 1993. – P. 252–256.

16. Friedman, S.J. Finding the optimal variable ordering for binary decision diagrams / S.J. Friedman, K.J. Supowit // IEEE Trans. Computers. – 1990. – Vol. 39, no. 5. – P. 710–713.

17. Fujii, H. Interleaving based variable ordering methods for ordered binary decision diagrams / H. Fujii, G. Ootomo, C. Hori // Computer-aided design : proc. of the 1993 IEEE/ACM Intern. conf., Santa Clara, 7–11 Nov. 1993 / IEEE Computer Society Press. – Los Alamitos, 1993. – P. 38–41.

18. Dynamic variable reordering for BDD minimization / E. Felt [et al.] // Design Automation : proc. EURO-DAC, Hamburg, 20–24 Sep. 1993 / IEEE Computer Society Press. – Los Alamitos, 1993. – P. 130–135.

19. Jeong, C. Computer-Aided Design of Digital Systems / C. Jeong // Department of Computer Science [Electronic resource]. – Mode of access : http://www1.cs.columbia.edu/~cs6861/sis/espresso-examples/ex. – Date of access : 20.03.2017.

20. Бибило, П.Н. Логическое проектирование дискретных устройств с использованием продукционно-фреймовой модели представления знаний / П.Н. Бибило, В.И. Романов. – Минск : Беларус. навука, 2011. – 279 с.

21. Поляков, А.К. Языки VHDL и VERILOG в проектировании цифровой аппаратуры / А.К. Поляков. – М. : СОЛОН-Пресс, 2003. – 320 с.

22. Лохов, А. Функциональная верификация СБИС в свете решений Mentor Graphics / А. Лохов // Электроника: наука, технология, бизнес. – 2004. – № 1. – С. 58–62.

Поступила 13.03.2017

Объединенный институт проблем информатики НАН Беларуси,
Минск, Сурганова, 6
e-mail: bibilo@newman.bas-net.by,
yurafreedom18@gmail.com

P.N. Bibilo, Y.Y. Lankevich

MINIMIZING THE MULTILEVEL REPRESENTATIONS OF SYSTEMS OF BOOLEAN FUNCTIONS BASED ON SHANNON DECOMPOSITION

A locally optimal algorithm is proposed to form a permutation of variables, which are used to obtain successive Shannon decompositions of a system of disjunctive normal forms of completely specified Boolean functions. The goal of it is a multilevel representation of functions which is called a reduced ordered Binary Decision Diagram. The results of experimental comparison of the program implementing the proposed algorithm and the program implementing the algorithm for enumeration of random permutations are given. The results show the advantage of the proposed algorithm when it is used for synthesis of logical circuits on the basis of library elements.