возможности минимальной. Элементы $B_i$ соответствуют каскадам в заданной схеме. Предлагается алгоритм получения решения данной задачи, близкого к оптимальному.

ЛИТЕРАТУРА


1. *Каган Б. М., Каневский М. М.* Цифровые вычислительные машины и системы. М.: Энергия, 1973.
2. *Кухарев Г. А., Шмерко В. П., Зайцева Е. Н.* Алгоритмы и систолические процессоры для обработки многозначных данных. Минск: Навука і тэхніка, 1990.
3. *Кнут Д.* Искусство программирования для ЭВМ. Т. 1. Основные алгоритмы. М.: Мир, 1976.


УДК 519.7

## АЛГОРИТМ ПОИСКА ЗАПРЕТОВ БУЛЕВЫХ ФУНКЦИЙ

Д. В. Рябоконь

Предложен алгоритм поиска запрета булевой функции, основанный на методе ветвей и границ и позволяющий находить некоторый запрет булевой функции, запрет минимальной длины или все запреты до заданной длины.

**Ключевые слова:** *запрет булевой функции, граф де Брёйна.*

Будем обозначать $P_2(n)$ множество всех булевых функций от $n$ переменных.

**Определение 1.** Говорят, что булева функция $f \in P_2(n)$ *имеет запрет* $y_1 \dots y_m \in \in \{0,1\}^m$, если система булевых уравнений

$$f(x_i, \dots, x_{n+i-1}) = y_i, \quad i = 1, \dots, m, \tag{1}$$

несовместна. Если для любых $m \in \mathbb{N}$ и $y_1 \dots y_m \in \{0,1\}^m$ система (1) совместна, то функция $f$ называется *функцией без запрета.*

**Определение 2.** *Графом де Брёйна* $G_n(f)$ функции $f \in P_2(n)$ называют граф, у которого вершины — это все булевы векторы длины $n-1$, а дуги поставлены во взаимно однозначное соответствие всевозможным булевым векторам длины $n$ так, что вектору $b_1 b_2 \dots b_n$ соответствует дуга, направленная от вершины $b_1 b_2 \dots b_{n-1}$ к вершине $b_2 b_3 \dots b_n$ и помеченная значением $f(b_1 b_2 \dots b_n)$.

Вершину графа $G_n(f)$ назовём *особой вершиной типа* $\delta \in \{0,1\}$, если обе исходящие из неё дуги помечены значением $\delta$. Будем говорить, что *вектор* $y_1 y_2 \dots y_n$ *является продолжением вектора* $x_1 x_2 \dots x_n$, если $y_1 \dots y_{n-1} = x_2 \dots x_n$.

**Утверждение 1.** В графе де Брёйна $G_n(f)$ функции $f \in P_2(n)$, $n > 1$, имеющей запрет, есть хотя бы одна особая вершина.

***Доказательство.*** Среди всех запретов функции $f$ выберем любой запрет минимальной длины $m$. Пусть это есть $e_1 \dots e_m$. Так как $m$ — минимальная длина запрета, граф $G_n(f)$ является $(m-1)$-полным графом де Брёйна, т. е. для любой последовательности $y_1 \dots y_{m-1} \in \{0,1\}^{m-1}$ существует путь, её несущий. Рассмотрим последовательность $e_1 \dots e_{m-1}$. Для неё в $G_n(f)$ есть несущий путь, пусть он оканчивается в вершине $v$. Так как $e_1 \dots e_m$ — запрет, из вершины $v$ нет дуги, помеченной $e_m$, т. е. обе исходящие из $v$ дуги имеют значение $\overline{e_m}$ и, следовательно, $v$ — особая вершина. ■

**Задача 1.** Предлагается следующий алгоритм *поиска некоторого запрета* функции $f \in P_2(n)$. Пусть $M_f^0$, $M_f^1$ — множества булевых векторов длины $n$, на которых функция $f$ принимает значения 0 и 1 соответственно; через $S(V)$ для $V \subseteq \{0,1\}^n$ будем обозначать множество всевозможных продолжений векторов из $V$.

Работу алгоритма можно представить как построение следующего двоичного дерева. Вершинам дерева соответствуют множества булевых векторов длины $n$, дугам — значения 0 и 1. Корню дерева сопоставим множество $\{0,1\}^n$. На каждом последующем шаге выбирается вершина ветвления, которой приписано множество $V$ наименьшей мощности, если таких вершин несколько, то выбираем любую из них. Из выбранной вершины порождаются два потомка: левому (правому) потомку приписывается множество $M_f^0 \cap S(V)$ (соответственно $M_f^1 \cap S(V)$) и ведущая к нему дуга помечается нулём (соответственно единицей).

Если некоторой вершине $u$ соответствует пустое множество, то запрет найден — это последовательность меток дуг пути из корня в вершину $u$.

Если вершине $u$ соответствует одноэлементное множество $\{x_1 \ldots x_n\}$, то проверяем, является ли вектор $x_2 \ldots x_n$ особой вершиной (типа $\delta$) в графе де Брёйна функции $f$, и если да — запрет найден, это $e_1 \ldots e_{m-1}\overline{\delta}$, где $e_1 \ldots e_{m-1}$ — последовательность меток дуг пути от корня до вершины $u$. Эта проверка легко выполняется по вектору значений функции $f$: вычисляется $h = f(x_2, \ldots, x_n, 0) \oplus f(x_2, \ldots, x_n, 1)$ и, если $h = 0$, то $x_2 \ldots x_n$ — особая вершина типа $f(x_2, \ldots, x_n, 0)$.

После каждого ветвления проверяем, есть ли среди полученных вершин неперспективные. Вершина объявляется неперспективной (и ветвление из неё не производится), если её метка (сопоставленное ей множество векторов) совпадает с меткой какой-либо вершины, встречавшейся ранее. Стоит заметить: это не значит, что при ветвлении из неперспективной вершины не будет найден запрет; это будет повторение одних и тех же действий. Для быстрого поиска вершины с заданной меткой будем хранить пройденные вершины в списке, индексированном по мощностям меток.

Если на каком-то шаге все вершины дерева объявлены неперспективными, а запрет ещё не найден, то у функции нет запрета.

По сравнению с алгоритмом 1, приведённым в [1], этот алгоритм более пригоден для практической реализации, так как в нём все шаги детально проработаны.

**Задача 2.** Для *поиска запрета минимальной длины* в исходном алгоритме нужно при нахождении запрета не завершать работу, а запомнить запрет и продолжать построение дерева таким же образом, пока не найдётся другой запрет; если он короче имеющегося, то запоминаем его. Дерево строится до тех пор, пока не исключим все вершины. При этом меняется критерий неперспективности вершины: вершина $v$, метка которой совпадает с меткой встречавшейся ранее вершины $u$, считается неперспективной в случае, если длина пути от корня до $v$ больше или равна длине пути от корня до $u$. Для этого в списке пройденных вершин наряду с меткой каждой вершины будем хранить длину пути от неё до корня. Если найден запрет длины $l$, то все вершины $l$-го яруса объявляются неперспективными.

**Задача 3.** Назовём запрет *тупиковым*, если никакое его подслово не является запретом. Задача *поиска всех тупиковых запретов до заданной длины* $l$ решается так же, как задача 2, но с запоминанием всех найденных запретов. Вершина объявляется неперспективной, если её метка совпадает с меткой предка (в этом случае можем получить запрет, но он не будет тупиковым), или она находится на расстоянии $l$ от корня.

Алгоритм поиска произвольного запрета был применён ко всем функциям от 2, 3 и 4 переменных; результаты приведены в табл. 1.

Таблица 1

**Количество функций, имеющих запрет**

| $n$ | Кол-во функций с запретом | Кол-во функций без запрета |
|---|---|---|
| 2 | 10 | 6 |
| 3 | 226 | 30 |
| 4 | 64954 | 582 |

В табл. 2 приведены данные о среднем времени работы алгоритма поиска произвольного запрета для функций с количеством аргументов больше 5.

Таблица 2

**Время работы алгоритма**

| $n$ | Среднее время, с | $n$ | Среднее время, с |
|---|---|---|---|
| 5 | 0,00019 | 13 | 0,128 |
| 6 | 0,0004 | 14 | 0,307 |
| 7 | 0,00091 | 15 | 0,698 |
| 8 | 0,0021 | 16 | 1,548 |
| 9 | 0,0047 | 17 | 3,461 |
| 10 | 0,011 | 18 | 7,552 |
| 11 | 0,025 | 19 | 16,528 |
| 12 | 0,061 | 20 | 33,312 |
| | | 21 | 77,999 |

В дальнейшем предполагается разработать и реализовать алгоритмы решения задач 1–3 с помощью обхода дерева в ширину и в глубину (поиск с возвратом) и сравнить трудоёмкости этих алгоритмов (по количеству построенных вершин дерева и сложности их обработки).

## ЛИТЕРАТУРА

1. *Колегов Д. Н.* О булевых функциях без запрета // Вестник Томского государственного университета. Приложение. 2005. № 14. С. 58–60.

УДК 519.7

# ОБ ЭФФЕКТИВНОМ ПРЕДСТАВЛЕНИИ ДИЗЪЮНКТИВНЫХ НОРМАЛЬНЫХ ФОРМ ДИАГРАММАМИ СПЕЦИАЛЬНОГО ВИДА[1]

А. А. Семенов

Рассматриваются диаграммы нового типа, используемые для представления дизъюнктивных нормальных форм. Показано, что данный класс диаграмм можно применять для компактного представления баз конфликтных дизъюнктов, накапливаемых в процессе нехронологического DPLL-вывода.

**Ключевые слова:** *ДНФ, решающие диаграммы, BDD, ZDD, дизъюнктивные диаграммы.*

[1]Работа выполнена при поддержке гранта РФФИ № 11-07-00377.