**Введение**

Тема контрольной работы «Математическая логика».

БУЛЬ или БУЛ, а также БУУЛ, Джордж (1815-1864) – английский математик, который считается основоположником математической логики.

Математическая логика – это раздел математики, посвященный анализу методов рассуждений, при этом в первую очередь исследуются формы рассуждений, а не их содержание, т.е. исследуется формализация рассуждений.

Формализация рассуждений восходит к Аристотелю. Современный вид аристотелева (формальная) логика приобрела во второй половине XIX века в сочинении Джорджа Буля “Законы мысли”.

Интенсивно математическая логика начала развиваться в 50-е годы XX века в связи с бурным развитием цифровой техники.

**1. Элементы математической логика**

Основными разделами математической логики являются исчисление высказываний и исчисление предикатов.

Высказывание – есть предложение, которое может быть либо истинно, либо ложно.

Исчисление высказываний – вступительный раздел математической логики, в котором рассматриваются логические операции над высказываниями.

Предикат – логическая функция от п переменных, которая принимает значения истинности или ложности.

Исчисление предикатов – раздел математической логики, объектом которого является дальнейшее изучение и обобщение исчисления высказываний.

Теория булевых алгебр (булевых функций) положена в основу точных методов анализа и синтеза в теории переключательных схем при проектировании компьютерных систем.

**1.1 Основные понятия алгебры логики**

Алгебра логики – раздел математической логики, изучающий логические операции над высказываниями.

В алгебре логики интересуются лишь истинностным значением высказываний. Истинностные значения принято обозначать:

1 (истина) 0 (ложь).

Каждой логической операции соответствует функция, принимающая значения 1 или 0, аргументы которой также принимают значения 1 или 0.

Такие функции называются логическими или булевыми, или функциями алгебры логики (ФАЛ). При этом логическая (булева) переменная *x* может принимать только два значения: .



Таким образом, - логическая функция, у которой логи-ческие переменные являются высказываниями. Тогда сама логическая функция является сложным высказыванием.



В этом случае алгебру логики можно определить, как совокупность множества логических функций с заданными в нем всевозможными логическими операциями. Таким логическим операциям, как конъюнкция (читается **И)**, дизъюнкция (**ИЛИ**), импликация, эквивалентность, отрицание (**НЕ)**, соответствуют логические функции, для которых приняты обозначения (&, ·), ~, – (), и имеет место таблица истинности:



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | *x~y* |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 |

Это табличный способ задания ФАЛ. Наряду с ними применяется задание функций с помощью формул в языке, содержащем переменные *x, y, …, z* (возможно индексированные) и символы некоторых конкретных функций – аналитический способ задания ФАЛ.

Наиболее употребительным является язык,содержащий логические символы ~, –. Формулы этого языка определяются следующим образом:



1) все переменные есть формулы;

2) если *P* и *Q* – формулы, то *P* ~ *Q*, - фор-мулы.



Например, выражение ~ - формула. Если переменным *x, y, z* придать значения из двоичного набора 0, 1 и провести вычисления в соответствии с операциями, указанными в формуле, то получим значение 0 или 1.



Говорят, что *формула реализует функцию.* Так формула ~ реализует функцию *h*(*x, y, z*):



|  |  |  |  |
| --- | --- | --- | --- |
| *x* | *y* | *z* | *h*(*x, y, z*) |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

Пусть *P* и *Q* – формулы, которые реализуют функции *f* (*x*1, *x*2, …, *xn*) и *g* (*x*1, *x*2, …, *xn*). Формулы равны: *P = Q*, если функции *f* и *g* совпадают, т.е. совпадают их таблицы истинности. Алгебра, основным множеством которой является все множество логических функций, а операциями – дизъюнкция, конъюнкция и отрицание, называется булевой алгеброй логических функций.

Приведем законы и тождества, определяющие операции – и их связь с операциями , ~:



1. Идемпотентность конъюнкции и дизъюнкции:

.



2. Коммутативность конъюнкции и дизъюнкции:

.



3. Ассоциативность конъюнкции и дизъюнкции:

.



4. Дистрибутивность конъюнкции относительно дизъюнкции и дизъюнкции относительно конъюнкции:

.



5. Двойное отрицание:

.



6. Законы де Моргана:

=, =.



7. Склеивание:

.



8. Поглощение

.



9. Действия с константами 0 и 1:

.



10. Законы Блейка-Порецкого:

.



11. Связь импликации с отрицанием – и дизъюнкцией :



.



12. Связь эквивалентности ~ с дизъюнкцией , конъюнкцией и отрицанием:



~ *y* =.



Всякая функция алгебры логики может быть реализована некоторой формулой языка с символами ~, –.



**1.2 Дизъюнктивные и конъюнктивные нормальные формы (ДНФ и КНФ)**



*ДНФ* и *КНФ* играют особую роль в алгебре логики и ее приложениях. Введем обозначение:



Так определенная переменная или ее отрицание называется *первичным термом.*

Формула вида, где - двоичный набор, а среди переменных нет одинаковых, называется элементарной конъюнкцией.



Всякая дизъюнкция элементарных конъюнкций называется дизъюнктивной нормальной формой (ДНФ):

.



Формула вида называется элементарной дизъюнкцией.



Всякая конъюнкция элементарных дизъюнкций называется конъюктивной нормальной формой (КНФ):

.



**Пример.**

Привести формулу ~*z* к *ДНФ* и *КНФ*.



1) Приведем формулу к *ДНФ* (последовательно: на основании определений операций импликации и эквивалентности, законов де Моргана и дистрибутивности):

~ ~(()=



.



2) Применив закон дистрибутивности к последнему выражению, получим *КНФ*:



Совершенной *ДНФ* (*СДНФ)* называется *ДНФ*, в которой нет равных элементарных конъюнкций и все элементарные конъюнкции содержат одни и те же переменные, причем каждую переменную – только один раз (включая вхождения под знаком отрицания).

Совершенная *КНФ* (*СКНФ)* определяется как такая *КНФ*, в которой нет одинаковых сомножителей; все сомножители содержат одни и те же переменные, причем каждую переменную – только один раз.

Для каждой ФАЛ можно построить реализующую ее *СДНФ*:



,



где дизъюнкция берется по тем двоичным наборам, на которых *f =* 1.

Каждая функция алгебры логики реализуется следующей *СКНФ*:



**Пример.**

Функция *h*(*x, y, z*), рассмотренная ранее, имеет следующую *СДНФ* (выписывается по единичным значениям) и *СКНФ* (выписывается по нулевым значениям):

1



0

;



|  |  |  |  |
| --- | --- | --- | --- |
| *x* | *y* | *z* | *h*(*x,y,z*) |
| 0 | 0 | 0 | 0 |
| *0* | *0* | *1* | *1* |
| 0 | 1 | 0 | 0 |
| *0* | *1* | *1* | *1* |
| 1 | 0 | 0 | 0 |
| *1* | *0* | *1* | *1* |
| *1* | *1* | *0* | *1* |
| 1 | 1 | 1 | 0 |

**Пример.**

Построить *СДНФ* и *СКНФ* будевой функции *f*(*x*1, *x*2, *x*3), заданной таблицей истинности

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *x*1 | *x*2 | *x*3 | *f*(*x*1,*x*2,*x*3) | *x*1 | *x*2 | *x*3 | *f*(*x*1,*x*2,*x*3) |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |



.



Разложение булевой функции по *k* переменным *x*1, *x*2,…, *xk* называется *разложением Шеннона.*



**1.3 Теорема Шеннона**

Любая булева функция представима в виде разложе-ния Шеннона:



где , - первичные термы.



**Пример.**

Пусть *п* = 4, *k* = 2. Тогда разложение Шеннона будет иметь вид



**Следствие.**

Предельное разложение Шеннона (*k* = *n*) булевой функции имеет вид



.



Предельное разложение Шеннона булевой функции является ее *СДНФ.*



В алгебре логики справедлив принцип двойственности. Согласно этому принципу, будем иметь следующие двойственные разложения Шеннона булевой функции :



по *k* переменным



двойственное предельное разложение

.



Двойственное предельное разложение Шеннона булевой функции является ее *СКНФ.*



**2 Булевы функции двух переменных**

Рассмотрим простой бинарный элемент – выключатель, который имеет два состояния. Если данный выключатель контролируется входной переменной *х*,то говорят, что он выключен (открыт) при *х* = 0 и включен (закрыт) при *х* = 1, как показано на рис. 1:



*х* = 0 *х* = 1

Рис. 1 - Два состояния выключателя

Будем использовать следующее графическое обозначение для представления таких выключателей:



*х*

Рис. 2 - Графическое обозначение выключателя

Рассмотрим соединения лампы с источником питания, представленные следующими схемами:

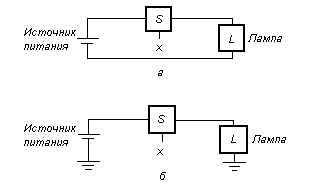


Рис. 3 - Лампа, управляемая выключателем: а – простое соединение с батареей; б – использование заземления, как обратной связи

Используя условное обозначение *L,* можно описать состояние лампы как функции входной переменной. Если лампа светится, то *L* = 1. Если лампа не светится, то *L* = 0. Поскольку *L* = 1 при *х* = 1, *L* = 0 при *х* = 0, то можно говорить, что *L*(*х*) = *х* – логическая функция, *х* – логическая переменная. Это простое логическое выражение описывает выход как функцию от входа.

**2.1 Булевы функции от двух переменных**

Рассмотрим теперь возможность использования двух выключателей для управления состоянием лампы. Пусть *х*1 и *х*2 – управляющие переменные (входы) для этих выключателей. Выключатели могут быть соединены последовательно или параллельно, как показано на рис. 4:

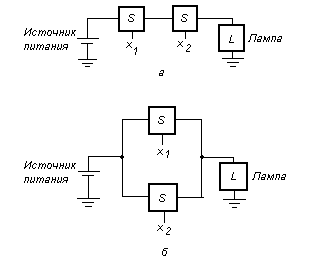


Рис. 4 - Две основные функции: а – последовательное соединение (функция логического умножения AND); б – параллельное соединение (функция логического сложения OR)

**2.2 Последовательное соединение двух выключателей**

При последовательном соединении лампа будет светиться только, если оба выключателя включены (одновременно). Это поведение может быть описано выражением: , где *L* = 1 при *х*1 = *х*2 = 1,



*L* = 0 в противном случае.

Символ называется AND-оператором. Говорят, что схема на рис. 3.4,а реализует логическую AND-функцию (логическое умножение).



**2.3 Параллельное соединение двух выключателей**

При параллельном соединении двух выключателей лампа будет гореть, если выключатели *х*1 и *х*2 включены. Лампа также будет гореть, если оба выключателя включены (одновременно). Лампа не будет гореть только, если оба выключателя открыты (разомкнуты, выключены). Это поведение может быть описано как:

, где *L* = 1 при *х*1 = 1 или *х*2 = 1, или *х*1 = *х*2 = 1; *L* = 0 при *х*1 = *х*2 = 0.



Символ называется OR-оператором. Говорят, что схема на рис. 4,б реализует логическую OR-функцию(логическое сложение).



В приведенных выше выражениях для AND и OR реализует результат (выход) есть логическая функция с двумя входными переменными. Функции AND и OR являются двумя наиболее важными логическими функциями. Вместе с некоторыми другими простыми функциями они могут быть использованы как составные части (строительные блоки) для реализации логических схем.



Например, на рис. 5 показано, как три выключателя могут быть использованы для управления лампой в более сложном случае. Такое последовательно-параллельное соединение выключателей реализует логическую функцию:

.



Лампа горит, если *х*3 = 1 и одновременно равны 1 либо *х*1, либо *х*2 (*х*1 = 1 или *х*2 = 1)

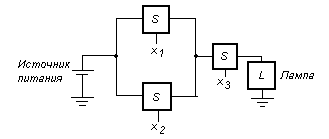


Рис. 5 - Последовательно-параллельное соединение выключателей

**2.4 Инверсия**

Пусть лампа подсоединяется к источнику питания так, как показано на рис. 6:

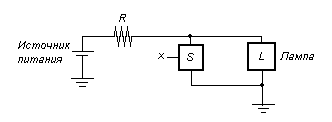


Рис. 6 - Инвертирующая схема

В этом случае выключатель соединяется параллельно с лампой. Лампа будет гореть, когда выключатель выключен. Формально такое функци- ональное поведение выражается: , где *L* = 1 при *х* = 0 и *L* = 0 при *х* = 1. Значение этой функции обратно значению входной переменной.



Вместо слова инверсия существует более общий термин отрицание.

Таким образом, есть отрицание *х*: . Символ ¯ называют NOT-оператором.



Количество логических функций в зависимости от числа переменных равно . Булевых функций одной переменной – четыре:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *x* | *f*0 | *f*1 | *f*2 | *f*3 |
| 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |

Индекс *I* функциональной переменной *fi*, *I* = 0, 1, 2, 3, равен десятичному эквивалентному набору значений этой функции, читаемому сверху вниз.

Функции *f*0(*x*) и *f*3(*x*) – константы 0 и 1 соответственно. Их значения не зависят от значения переменной *х*. Функция *f*1(*x*) “ повторяет “ *х*: *f*1(*x*) = *x*.

Функция *f*2(*x*) называется *отрицанием х* (или функцией НЕ) и обозначается , т.е. *f*2(*x*) = . Ее значение противоположно значению *х*.



Булевых функций двух переменных – шестнадцать:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x*1 | *x*2 | *f*0 | *f*1 | *f*2 | *f*3 | *f*4 | *f*5 | *f*6 | *f*7 | *f*8 | *f*9 | *f*10 | *f*11 | *f*12 | *f*13 | *f*14 | *f*15 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

Индекс *I* функциональной переменной *fi*, *I* = 0, 1, 2, …, 15, равен десятичному эквивалентному набору значений этой функции, читаемому сверху вниз. Приведем эти булевы функции:

- константа ноль;



- конъюнкция;



*х*1 –|→ - левая коимпликация (читается «*не если х*1, *то х*2», префикс *ко* – от лат. conversus – *обратный)*;



;



*х*1 ←|– *х*2 - правая коимпликация;



;



- сложение по модулю два или функция неравнозначности, неэквивалентности;



- дизъюнкция*;*



*х*1 ◦ *х*2 = *х*1 ↓ *х*2 - функция Вебба(*Пирса*);



*х*1 ~ *х*2 –функция эквивалентности, равнозначности;



- отрицание *х*2;



- правая импликация (читается « *если х*2, *то х*1»);



- отрицание *х*1;



- левая импликация (читается «*если х*1, *то х*2»);



- функция Шеффера;



- константа единица.



**2.5 Свойства элементарных функций алгебры логики**

**2.5.1 Функция сложения по модулю два (по mod 2)**

Пусть Операция “сложениe по mod *p* “ определяется следующим образом: *а* *b* = *c*, где *с* – остаток от деления на *p* числа *a* + *b*. Например, если *р* = 7, то . Тогда , .



При сложении по mod 2: *р* = 2, . Тогда при *а* = *х*1, *b* = *x*2 получим:



|  |  |  |
| --- | --- | --- |
| *х*1 | *х*2 |  |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

.



Справедливы коммутативный и ассоциативный законы. Дистрибутивный закон имеет вид:



Аксиомы:



Связь с функциями ¯:



.



**2.5.2 Функция Вебба (Пирса)**



Аксиомы: .



Коммутативность: .



Формулы преобразования функций ¯ через ↓:



.



**2.5.3 Функция импликации**



Аксиомы: .



Импликация обладает свойством коммутативности в виде:

;



ассоциативность не выполняется.

Формулы преобразования функций ¯ через →:



.



**2.5.4 Функция Шеффера**



Аксиомы: .



Свойство коммутативности верно только для двух переменных:



ассоциативность не выполняется.

Формулы преобразования:



**3. Системы функций алгебры логики**

**3.1 Функциональная полнота**

Алгебра над множеством логических функций с двумя бинарными операциями и называется *алгеброй Жегалкина.* В алгебре Жегалкина выполняются следующие соотношения:



а также соотношения булевой алгебры, относящиеся только к конъюнкции и константам (с конъюнкцией).

Отрицание и дизъюнкция в этой алгебре выражаются следующим образом:



Формулы, содержащие знаки называют *полиномами.*



Полином вида: , где есть либо 1, либо переменная, либо конъюнкция различных переменных, при , называется *полиномом Жегалкина.*



**Теорема.**

Любая булева функция может быть представлена полиномом Жегалки- на



где *ki* – коэффициенты, принимающие значения 0 или 1: .



**3.2 Класс линейных функций (К Л)**

Булева функция называется *линейной*, если она представима полиномом первой степени

.



Количество линейных функций равно , где *п* – число перемен-ных.



Для *п* = 2 их 8:



**3.3 Класс функций, сохраняющих ноль (К 0)**

Если булева функция на нулевом наборе переменных равна нулю, то говорят, что *функция сохраняет ноль:*



**3.4 Класс функций, сохраняющих единицу (К 1)**

Если булева функция на единичном наборе переменных равна единице, то говорят, что *функция сохраняет единицу*:



**3.5 Класс монотонных функций (К м)**

Булева функция называется монотонной, если для любых двоичных наборов из того, что , выполняется неравенство: .



**3.6 Класс самодвойственных функций (К с)**

Булева функция называется *двойственной* для функции , если таблицу истинности для функции можно получить из таблицы для функции *f*, заменив в значениях аргумента функции 0 на 1 и 1 на 0, т.е. имеет место равенство



Например, конъюнкция и дизъюнкция двойственны друг другу, отрицание двойственно самому себе.

Функция, совпадающая со своей двойственной, называется *самодвойственной.*

Самодвойственная функция на противоположных наборах и принимает противоположные значения. Противоположными являются наборы, сумма десятичных эквивалентов которых равна 2*n* – 1, где *п* – количество переменных, от которых зависит функция.



Распределение булевых функций двух переменных по классам

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Функция | К л | К 0 | К 1 | К м | К с |
| *f* 0 | \* | \* |  | \* |  |
| *f* 1 |  | \* | \* | \* |  |
| *f* 2 |  | \* |  |  |  |
| *f* 3 | \* | \* | \* | \* | \* |
| *f* 4 |  | \* |  |  |  |
| *f* 5 | \* | \* | \* | \* | \* |
| *f* 6 | \* | \* |  |  |  |
| *f* 7 |  | \* | \* | \* |  |
| *f* 8 | - | - | - | - | - |
| *f* 9 | \* |  | \* |  |  |
| *f* 10 | \* |  |  |  | \* |
| *f* 11 |  |  | \* |  |  |
| *f* 12 | \* |  |  |  | \* |
| *f* 13 |  |  | \* |  |  |
| *f* 14 | - | - | - | - | - |
| *f* 15 | \* |  | \* | \* |  |

**3.7 Принцип двойственности**

Если в формуле алгебры логики *F* заменить знаки всех логических функций на знаки двойственных функций, то получится двойственная формула *F* \*, реализующая функцию, двойственную той, которая реализуется формулой *F*. При этом, если формулы равны *F* 1 = *F* 2, то верно равенство двойственных формул , которое называется *двойственным* предыдущему. Например, равенства, являющиеся двойственными друг другу:



и ;



и ;



и ;



и ;



и .



**3.8 Полнота функций алгебры логики**

*Суперпозицией функций*  называется функция *f*, полученная с помощью подстановок этих функций друг в друга и переименования переменных, а *формулой* называется выражение, описывающее эту суперпозицию.



Например, суперпозицию функций *f*1, *f*2, *f*3 можно задать формулой

.



Если *f*1 обозначает дизъюнкцию, *f*2 – конъюнкцию, а *f*3 – сложение по mod 2, то последняя формула примет вид:

.



Если рассматривается произвольная система функций, то возникает вопрос: всякая ли логическая функция из этой системы представима формулой, содержащей символы только этой системы функций.

Система функций алгебры логики (ФАЛ) называется *функционально полной*, если любая функция алгебры логики может быть реализована формулой, содержащей лишь символы функций из этой системы ФАЛ, т.е. является суперпозицией функций из этой системы.

Следовательно, система функций должна быть функционально полной.

**3.9 Теорема Поста – Яблонского (критерий функциональной полноты)**

Для того, чтобы система ФАЛ была полной необходимо и достаточно, чтобы она содержала функцию:



1) не сохраняющую ноль;

2) не сохраняющую единицу;

3) нелинейную;

4) немонотонную;

5) несамодвойственную.

Примерами функционально полных систем являются, например, системы:

.



Все названные выше классы функций обладают свойством: любая ФАЛ, полученная с помощью операции суперпозиции и подстановки из функций одного класса, обязательно будет принадлежать этому же классу.

Полная система ФАЛ называется *базисом*,если теряется полнота Ф при удалении хотя бы одной функции системы.

К базису относятся системы функций:

базис 1: ;



базис 2: ;



базис 3: ;



базис 4: функция Шеффера: *x*1 | *x*2;

базис 5: функция Пирса (Вебба): *x*1 ↓ *x*2.

Базис 1 – избыточный, базисы 4 и 5 – минимальные (удаление хотя бы одной функции превращает систему ФАЛ в неполную).

При исследовании полноты систем функций удобно пользоваться таблицей, которую называют *критериальной*. Эта таблица имеет пять столбцов, каждый из которых соответствует одному из пяти классов, а строки таблицы соответствуют функциям исследуемой системы. На пересечении строки таблицы, соответствующей функции *f*, и столбца, соответствующего классу *К*, ставится знак плюс, если функция , и минус, если . Система функций полна тогда и только тогда, когда в каждом столбце содержится хотя бы один знак минус.



**Пример.**

Является ли система булевых функций полной? Если является, то выписать все возможные базисы.



Рассмотрим функцию .



1. Исследуем ее принадлежность к классу *К*0:

.



Следовательно, .



2. Исследуем принадлежность функции к классу *К*1:

.



Следовательно, .



3. Установим, является ли функция *f*1 линейной. Используем метод неопределенных коэффициентов. Предположим, что функция линейная и, следовательно, представима в виде полинома Жегалкина первой степени:

.



Найдем коэффициенты , исходя из предположения линейности этой функции. Зафиксируем набор 000:



, , .



Следовательно, .



Зафиксируем набор 100:

,



,



.



Следовательно, .



Фиксируем набор 010:

,



Фиксируем набор 001:



.



Следовательно, функция (по нашему предположению) может быть представлена полиномом первой степени вида:

.



Если функция линейная, то полученный полином, путем тождественных преобразований, должен привестись к виду заданной функции. Ясно, что полученный полином не приводится к исходной функции. Следовательно, .



4. Исследуем заданную функцию на самодвойственность.

Функция самодвойственная, если на любой паре противоположных наборов (наборов, сумма десятичных эквивалентов которых равна , где *п* – количество переменных функции) функция принимает противоположные значения.



Построим таблицу: ; вычислим значения функции на оставшихся наборах:



:



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| (000)  0 | (001)  1 | (010)  2 | (011)  3 | (100)  4 | (101)  5 | (110)  6 | (111)  7 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |

На наборах 0 и 7, 1 и 6 функция принимает одинаковые значения. Следовательно .



5. Проверим принадлежность заданной функции *f*1 классу монотонных функций. Из таблицы видно: 001< 010, но . Следовательно, функция .



Рассмотрим функцию .



1. Принадлежность функции классу *К*0:

.



Следовательно, .



2. Принадлежность функции классу *К*1:

.



Следовательно, .



3. Принадлежность функции классу *К л*.

Предполагаем, что

.



Фиксируем набор 0000:

,



, .



Фиксируем набор 1000:

,



.



Фиксируем набор 0100:

,



.



Фиксируем набор 0010:

,



.



Фиксируем набор 0001:

.



.



Окончательно получаем

.



Это равенство на других 11 наборах не выполняется. Действительно, для набора 1111 имеем

, , т.е. .



Следовательно, .



4. Принадлежность функции классу самодвойственных функций.

Строим таблицу:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |

Из таблицы видно, что на наборах 1 и 14, 2 и 13, 7 и 8 функция принимает одинаковые значения. Следовательно, .



5. Принадлежность функции классу монотонных функций.

Из таблицы видно, что 1000>0000, а . Следовательно, .



Строим критериальную таблицу:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | *К* 0 | *К* 1 | *К л* | *К с* | *К м* |
| *f* 1 | + | - | - | - | - |
| *f* 2 | - | - | - | - | - |

В таблице в каждом столбце стоит хотя бы один минус. Следовательно, система булевых функций является полной.



Найдем все возможные базисы. По критериальной таблице составим к.н.ф. *К*, в которой элементарные дизъюнкции соответствуют столбцам таблицы и включают в качестве слагаемых символы тех функций, которые не входят в класс, соответствующий столбцу. В данном случае имеем

.



Используя законы и свойства дизъюнкции и конъюнкции, приведем к.н.ф. *К* к д.н.ф. *D*, в которой упрощение невозможно. В нашем случае получим



.



По полученной д.н.ф. *D* выпишем подмножества функций, соответствующие слагаемым д.н.ф. *D*. Это и будут искомые базисы. В нашем случае имеется два базиса:

.



*Минимальная форма* представления ФАЛ содержит минимальное количество термов и переменных в термах (т.е. не допускает никаких упрощений).

**Пример.**

,



- минимальная форма.



**4 Способы представления функций алгебры логики**

Для булевых функций существуют различные способы представления: таблица истинности, числовой, аналитический**.**

**Пример.**

Пусть функция задана *таблицей истинности*:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № набора | *х*1 | *х*2 | *х*3 |  |
| 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 1 | 1 | 1 |
| 4 | 1 | 0 | 0 | 1 |
| 5 | 1 | 0 | 1 | 0 |
| 6 | 1 | 1 | 0 | 0 |
| 7 | 1 | 1 | 1 | 0 |

В таблице в первом столбце под номером набора понимается десятичный эквивалент соответствующего двоичного набора.

Тогда запись или является *числовым* представлением этой функции, где в скобках показаны номера наборов, на которых функция принимает значение 1.



Запись



представляет собой *аналитическое* представление булевой функции.

Можно в выражении функции вместо конъюнкций термов записать соответствующие двоичные наборы. Тогда получим:

.



Булеву функцию можно задать с помощью *единичного п – мерного куба* (рис. 7).

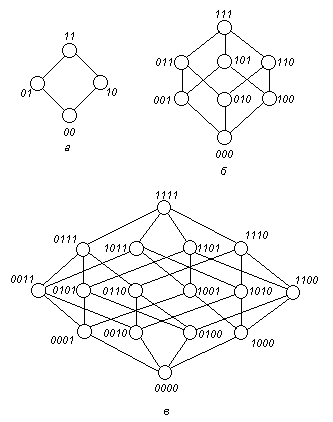


Рис. 7

*Единичным п – мерным кубом* называется граф, каждая вершина которого взаимно однозначно соответствует двоичному набору. Две вершины соединены ребром, если соответствующие наборы отличаются только в одном разряде (в одной координате).

На рис. 7 показаны *п-*мерные кубы для булевых функций: двух переменных (*а*), трех переменных (*б*), четырех переменных (*в*).

Отметив вершины, в которых булева функция принимает единичные значения, получим геометрическое представление булевой функции. Так функция



примет вид:

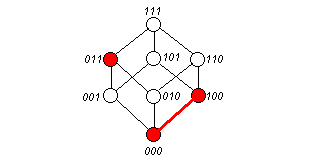


Рис. 8

Часто для изображения булевых функций двух и трех переменных используют прямоугольную систему координат:

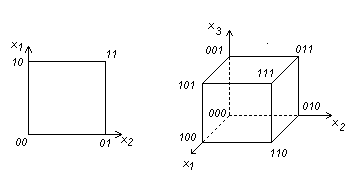


Рис. 9

Изображение булевых функций числа переменных более трех в этом случае невозможен.

В методе минимизации булевых функций Квайна – Мак-Класки используется кубическое представление булевых функций (аналог *п-*мерных кубов).

Терм максимального ранга принято называть 0-кубом и обозначать *К* 0. Если два 0-куба из комплекса *К* 0 различаются только по одной координате, то они образуют куб (отрезок) *K* 1.

**Пример.**

Для

.



**5. Минимизация булевых функций**

*Импликантой функции* называют некоторую логическую функцию, которая обращается в ноль на том же наборе переменных, на котором равна нулю сама функция.

Любой конъюнктивный терм (элементарная конъюнкция) или группа термов, соединенных знаками дизъюнкции, входящие в *СДНФ*, являются импликантами исходной функции

Элементарная конъюнкция (конъюнктивный терм), в которой удален один или несколько первичных термов, называется *простой импликантой.*

Методов минимизации булевых функций в настоящее время существует довольно много. Все они, как правило, состоят из двух этапов. На первом этапе получают список всех простых импликант, т.е. *сокращенную* *ДНФ*. На втором этапе, используя таблицу покрытий, удаляют “лишние” импликанты, покрывемые другими импликантами. В результате получают *ДНФ*, из которой нельзя удалить ни одной импликанты. Такую *ДНФ* называют *тупиковой.*

**5.1 Метод Квайна**

На первом этапе в методе Квайна попарно сравнивают все импликанты, входящие в *СДНФ*, в целях выявления возможности поглощения какой-то переменной на основе закона склеивания:

.



Процедура продолжается до тех пор, пока не останется ни одного члена, допускающего поглощение с другим термом. В результате получают некоторое количество простых импликант. Дизъюнкция этих импликант является сокращенной *ДНФ*.

На втором этапе строится таблица покрытий.В строках этой таблицы указываются найденные простые импликанты, а в столбцах – термы исходного выражения функции. Клетки таблицы отмечаются, если простая импликанта входит в состав какого-либо терма. В итоге минимизация булевой функции сводится к тому, чтобы найти такое минимальное количество простых импликант, которые покрывают все столбцы. В результате получают тупиковую *ДНФ*.

Недостаток метода–необходимость попарного сравнении всех конъюнктивных термов на первом этапе при нахождении простых импликант. С ростом числа исходных термов увеличивается количество попарных сравнений, что усложняет решение задачи минимизации.

**5.2 Метод Квайна с применением *п-*мерных кубов**

Данный метод устраняет недостаток предыдущего метода, т.е. устраняет необходимость попарного сравнения всех термов на первом этапе при нахождении простых импликант. Для этого строится *п-*мерный куб, по которому визуально можно определить те конъюнктивные термы, склеивание которых порождают простые импликанты.

При решении задачи минимизации булевой функции удобно вместо конъюнктивных термов использовать, соответствующие им, двоичные наборы.

**Пример.**

Минимизировать булеву функцию

.



Здесь в скобках стоят десятичные эквиваленты соответствующих двоичных наборов.

Представим функцию в виде *СДНФ*:



Запишем конъюнктивные термы в виде двоичных наборов:

.



Построим единичный 4 – мерный куб и выделим вершины, соответствующие двоичным наборам, входящим в заданную булеву функцию (рис. 10):

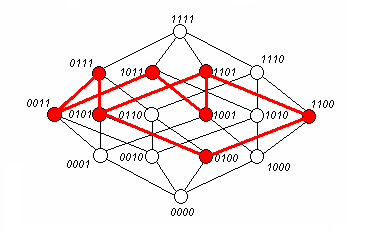


Рис. 10

**1 этап.** Определение сокращенной *ДНФ*.

Применим закон склеивания для отмеченных вершин (для двоичных наборов) куба, соединенных ребром:



Прочерк означает, что переменная в этом месте отсутствует.

Для некоторых простых импликант склеивание можно продолжить:



По закону идемпотентности:



Дизъюнкция полученных простых импликант является сокращенной ДНФ:



**2 этап.** Определение тупиковой *ДНФ*.

Для определения тупиковой *ДНФ* построим таблицу покрытий, в которую следует включать и двоичные наборы, не участвовавшие в склеивании:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Простые импликанты | Исходные термы | | | | | | | |
| 0011 | 0100 | 0101 | 0111 | 1001 | 1011 | 1100 | 1101 |
| 0 – 11 | \* |  |  | \* |  |  |  |  |  |
|  | \* |  |  |  |  | \* |  |  |
|  |  |  | \* | \* |  |  |  |  |
| 10 – 1 |  |  |  |  | \* | \* |  |  |
| 1 – 01 |  |  |  |  | \* |  |  | \* |
|  |  | \* | \* |  |  |  | \* |  |

Определяя минимальное число строк, покрывающих все столбцы таблицы, находим тупиковую *ДНФ*:



Недостатком методаявляется само построение *п* – мерного куба, т.к. при числе переменных это построение затруднительно.



**5.3 Метод Квайна – Мак-Класки**

Метод Квайна – Мак-Класки представляет собой предыдущий метод, но без геометрического построения *п* – мерных кубов: кубы присутствуют, но абстрактно.

Метод основан на кубическом представлении конъюнктивных термов *ДНФ* с предварительным разбиением кубов на подгруппы, определяемые одинаковым числом единиц. Разбиение дает возможность сравнивать кубы только соседних по числу единиц групп для уменьшения количества переборов.

В итеративной процедуре минимизации попарное сравнение можно производить только между соседними группами.

Нахождение простых импликат на первом этапе:

1. Все исходные конъюнктивные термы записываются в виде их двоичных наборов.

2. Все наборы разбиваются на непересекающиеся группы по числу единиц. Условие образования *r*-куба – наличие расхождения в (*r*-1)-кубах только по одной координате в одном двоичном разряде и наличие общих независимых координат.

3. В *i*-группу включают все двоичные наборы, имеющие в своей записи *i* единиц.

4. Попарное сравнение производится только между соседними по номеру группами. Группы, которые различаются в двух разрядах и более, не имеет смысла сравнивать.

**Пример.**

(Предыдущий пример)

Минимизировать булеву функцию



**1 этап.** Определение сокращенной *ДНФ*.

По двоичным наборам запишем 0-кубы:

*К* 0= {0011, 0100, 0101, 0111, 1001, 1011, 1100, 1101}.

Выполним их разбиение на подгруппы по количеству единиц:



Строим *К* 1-кубы, сравнивая соседние подгруппы:



Выполним разбиение всех *К* 1-кубов в зависимости от положения независимой координаты *Х*:



Выполним сравнение кубов внутри каждой подгруппы с целью построения *К* 2-кубов:



Поскольку они одинаковы, то



Записываем сокращенную *ДНФ*, в которую входят простые импликанты из *К* 1 (не вошедшие в *К* 2) и *К* 2:



**2 этап.** Определение тупиковой *ДНФ*.

Строим таблицу покрытий, в которую следует включать и те двоичные наборы, которые на любой итерации не участвовали ни разу в склеивании:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Простые  импликанты | Исходные термы | | | | | | | |
| 0011 | 0100 | 0101 | 0111 | 1001 | 1011 | 1100 | 1101 |  |
| 0 – 11 | \* |  |  | \* |  |  |  |  |
| - 011 | \* |  |  |  |  | \* |  |  |
| 01 – 1 |  |  | \* | \* |  |  |  |  |
| 10 – 1 |  |  |  |  | \* | \* |  |  |
| 1 – 01 |  |  |  |  | \* |  |  | \* |
| - 10 - |  | \* | \* |  |  |  | \* | \* |

Определяя минимальное число строк, покрывающих все столбцы таблицы, находим тупиковую *ДНФ*:



**Литература**

1. Горбатов В.А. Основы дискретной математики. – М.: Высшая школа, 1986. – 311 с.

2. Коршунов Ю.М. Математические основы кибернетики. – М.: Энергоатомиздат, 1987. – 496 с.

3. Кузнецов О.П., Адельсон-Вельский Г.М. Дискретная математика для инженера. – М.: Энергоатомиздат, 1988. – 480 с.

4. Сигорский В.П. Математический аппарат инженера. – М.: Техника, 1975. – 768 с.

5. Шапорев С.Д. Дискретная математика. – СПб., 2006. - 400 с.

6. Хаханов В.И., Чумаченко С.В. Дискретная математика. Конспект теоретического материала (электронная версия). ХНУРЭ, 2003.

7. Яблонский С.В. Введение в дискретную математику. – М.: Наука, 1979. – 272 с.

8. Гаврилов Г.П., Сапоженко А.А. Задачи и упражнения по дискретной математике. – М.: ФИЗМАТЛИТ, 2005. – 416 с.