Практическая работа №4. Алгебра логики и ее применения

Теоретическое введение

Отправным пунктом здесь должно быть осознание того, что буквами в булевой алгебре обозначены не вещи, а характеристики (определения, атрибуты) вещей. Другими словами, объекты отображаемой действительности представлены в булевой алгебре не непосредственно сами собой и даже не их обозначениями (именами), а обозначениями и выражениями присущих им качеств — их атрибутами.

Подобно тому как обычная числовая алгебра оперирует с величинами, т.е. количественными характеристиками вещей, оцениваемыми в числах, булева алгебра занимается качественными характеристиками — атрибутами, для точной оценки которых достаточно двух значений: 1 — «есть» («дан», «имеет место») и 0 — «нет» («исключен», «не имеет места»). Как отмечал основоположник развития алгебры логики в России П.С. Порецкий, обыкновенная алгебра — это алгебра количеств, а булева — алгебра качеств.

Базисные операции

Как математический объект булева алгебра вполне аналогична послужившей ей прототипом числовой алгебре, но отличается от нее набором базисных операций и множеством значений, на котором они определены. В соответствии с концепцией Буля это множество содержит лишь нулевой и единичный элементы. Базисных операций всего три: одна одноместная (однооперандная, монарная) и две двухместные (бинарные). Принципы организации и функционизования, свойственные числовой алгебре, полностью сохранены: основная конструкция (выражение) строится при помощи скобок (или эквивалентных правил бесскобочного синтаксиса) путем применения операций к ранее образованным тем же путем выражениям, которыми на каком-то этапе этого пути оказываются не детализируемые далее, *исходные*, символы, называемые обычно терминальными, или терминалами.

Например, выражение (a + 2)/(b - c/3) представляет собой произведение выражений a+2 и b - c/3, первое из которых построено непосредственно из терминальных символов a, 2, второе же — из терминала *b* и выражения c/3.

Выражение, входящее в составное выражение в качестве его под- выражения, называют *вложенным* в него. Вложенность естественно приводит к иерархии и рекурсии.

Одноместную операцию булевой алгебры сами алгебраисты называют *дополнением*, а логики и другие пользователи этой алгебры — *отрицанием*. О причинах и последствиях этого разногласия будет сказано в дальнейшем.

Двухместная мультипликативная (типа умножения) операция в булевой алгебре называется конъюнкцией, или логическим умножением, двухместная аддитивная (типа сложения) — дизъюнкцией, а иногда логическим сложением.

В качестве знака дополнения-отрицания применяются: штрих, черта над операндом, а также префиксы с и " минус ". Так, выражение "дополнение x " в зависимости от используемых обозначений может быть представлено в следующих вариантах: x’, -x, !x, x. Впрочем, не исключены и какие-нибудь иные.

Знаком дизъюнкции, как правило, служит ∨ (первая буква латинского *vel* — "неразделительное или"). Конъюнкцию обозначают этой же буквой, перевернутой верхом

вниз — ∧, а иногда знаком & либо "точкой". Нередко знак конъюнкции просто опускают, подобно знаку умножения в числовой алгебре. Поэтому возможны следующие разновидности выражения "конъюнкция x, y": x∧y, x·y, xy.

Исчерпывающим определением перечисленных базисных операций служит таблица:

x	**y**	**x'**	**x ∧ y**	**x ∨ y**
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

(X, Y)

x **y**	**0**	**1**
0	(0, 0)	(0, 1)
1	(1, 0)	(1, 1)

x'

0	**1**
1	1
0	0

x ∧ y

0	**1**
0	0
0	1

x ∨ y

0	**1**
0	1
1	1

Все эти определения носят, конечно, чисто арифметический характер. Например, определение конъюнкции является ни чем иным, как таблицей умножения чисел 0, 1. Алгебра же начинается с выявления присущих операциям свойств с целью установить законы тождественного преобразования выражений.

Основные тождества

В общем случае выражение состоит из знаков операций, букв и цифр, обозначающих те объекты, к которым применяются операции, и скобок, регламентирующих вложенность. В частности, выражением является и отдельная буква или цифра (а в иных системах и отдельный знак операции).

Вместе с тем, всякое выражение, подобно отдельной букве, может быть объектом операции (операндом) и так же как буква способна принимать значение из допустимых в булевой алгебре 0 и 1. Различие в том, что букве ее текущее значение придают непосредственно (присваивают). а значение выражения вычисляется, или оценивается, путем выполнения всех предписанных в нем операций над текущими значениями входящих в него букв и подвыражений в порядке их вложенности. Впрочем, можно и обратно, фиксировав значение выражения, устанавливать обеспечивающие его наборы значений букв.

Выражение, содержащее k различных букв, может быть вычислено на 2^k^ различных наборах значений этих букв и сопоставляет таким образом каждому из них (принимает на нем) однозначно определенное значение. Два выражения с одним и тем же набором букв называются тождественными (выражающими одно и то же). если на любом из наборов значений, присвоенном их буквам, значения этих выражений совпадают (одинаковы, равны).

Тождественность, или тождество, выражений принято обозначать знаком тождества ≡, который, будучи помещен между двумя выражениями, свидетельствует, что они тождественны друг другу. Нетождественность обозначается перечеркиванием знака тождества. Например: x≡x, x≠x', x≠y. Всякая буква предполагается тождественной сама себе и не тождественной любой другой букве.

Из таблиц, которыми определены базисные булевы операции, непосредственно устанавливаются следующие фундаментальные тождества, в совокупности достаточные для воспроизведения этих таблиц, т. е. составляющие эквивалентное определение операций:

- (x')' ≡ x — закон двойного отрицания-дополнения,
- x ∧ x ≡ x — идемпотентность конъюнкции,
- x ∨ x ≡ x — идемпотентность дизъюнкции,
- x ∧ x' ≡ 0 — закон противоречия,
- x ∨ x' ≡ 1 — закон исключенного третьего.

Определения операций и осуществляемое путем вложенности конструирование из них трехбуквенных выражений позволяют установить также ряд других законов булевой алгебры:

- x∧y ≡ y∧x, x∨y ≡ y∨x — переместительность (коммутативность) для конъюнкции и дизъюнкции,
- (x∧y)∧z ≡ x∧(y∧z), (x∨y)∨z ≡ (x∨y)∨z — сочетательность (ассоциативность) для конъюнкции, а также дизъюнкции,
- x(y∨z) ≡ xy∨xz, x∨yz ≡(x∨y)(x∨z) — распределительность (дистрибутивность) конъюнкции относительно дизъюнкции и дизъюнкции по отношению к конъюнкции,
- x(x∨y) ≡ x, x ∨xy ≡ x, x∧1 ≡ x, x∨0 ≡ x, x∧0 ≡ 0, x∨1 ≡ 1 — законы «поглощения»,
- (xy)' ≡ x' ∨ y', (x∨y)' ≡ x'y' — законы де Моргана.

Законы де Моргана, сводящие отрицание конъюнкции к дизъюнкции отрицаний, а отрицание дизъюнкции — к конъюнкции отрицаний, позволяют выразить конъюнкцию в терминах дизъюнкции и отрицания и обратно — дизъюнкцию через конъюнкцию и отрицание:

- xy ≡ (x' ∨ y')'

- $x \vee y \equiv (x'y')'$

Это свидетельствует об избыточности базисного набора операций булевой алгебры: для произвольного булева выражения существует тождественное ему, построенное без применения конъюнкции, а также тождественное и не содержащее дизъюнкции. Другими словами, функциональная полнота обеспечивается уже сочетанием отрицания и конъюнкции или отрицания и дизъюнкции. Однако только объединением этих систем достигаются присущие булевой алгебре естественность, гибкость и элегантность.

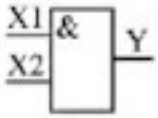
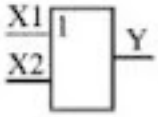
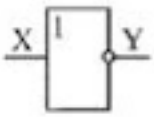
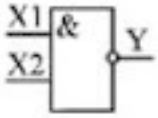
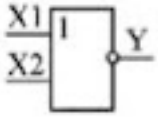
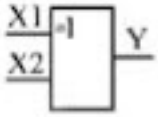
Совместным применением конъюнкции и дизъюнкции обусловлена свойственная булевой алгебре двойственность выражений. Сами операции конъюнкции и дизъюнкции называются двойственными (дуальными) друг другу. Двойственность понимается как "то же, но в обратном порядке", причем речь идет об упорядоченности множества значении, на котором определены операции.

Конъюнкция и дизъюнкция формируют свои значения единообразно, а именно выбором из значений, присвоенных их операндам. Разница только в том, что конъюнкция предпочитает 0, а дизъюнкция предпочитает 1. Так, значением конъюнкции будет 0, если хотя бы один ее операнд принял значение 0, значение же 1 она имеет лишь в случае, когда оно принято обоими операндами. У дизъюнкции все наоборот. Иначе говоря, дизъюнкция так поступает с 1, как конъюнкция с 0, а точнее: по отношению к последовательности 01 конъюнкция идентична дизъюнкции относительно обратной последовательности — 10. И та и другая отдает в своей последовательности предпочтение первому элементу.

Как операции над числами конъюнкция и дизъюнкция являются функциями, доставляющими соответственно минимальное и максимальное из значений своих аргументов:

- $xy \equiv \min(x, y)$,
- $x \vee y \equiv \max(x, y)$.

Задание 1. Схемотехническое применение

Название элемента	Условное обозначение элемента	Таблица истинности X2	X1	Y	Условное обозначение логической операции
2И		0	0	0	X1*X2 X1∧X2
		0	1	0	
		1	0	0	
		1	1	1	
2ИЛИ		0	0	0	X1+X2 X1∨X2
		0	1	1	
		1	0	1	
		1	1	1	
НЕ			0	1	$\overline{X}$ ¬X
			1	0	
2И-НЕ		0	0	1	$\overline{X1*X2}$ ¬(X1∧X2)
		0	1	1	
		1	0	1	
		1	1	0	
2ИЛИ-НЕ		0	0	1	$\overline{X1+X2}$ ¬(X1∨X2)
		0	1	0	
		1	0	0	
		1	1	0	
Исключающее ИЛИ		0	0	0	X1⊕X2
		0	1	1	
		1	0	1	
		1	1	0	

Варианты

1. Функция голосования «двое из трёх». Принимает три переменные и возвращает один результат, ориентируясь на выводы
2. Преобразователь трехразрядного двоичного кода в код Грея, разряд номер 0 (младший разряд)
3. Преобразователь трехразрядного двоичного кода в код Грея, разряд номер 1 (средний разряд)
4. Преобразователь трехразрядного двоичного кода в код Грея, разряд номер 2 (старший разряд)

5. Выбор максимума из двух чисел `A_0 A_` (`A < B`, `A = B`, `A > B`)
6. Функция попарного равенства четырёх переменных (`A = B` и `C = D`)
7. Одноразрядный сумматор (суммирует `A`, `B` и P_0 — выход суммы `S`.
8. Одноразрядный сумматор (суммирует `A`, `B` и P_0 — выход переноса `P`.

1а. Построение таблицы истинности

Как мы убедились из введения, таблица истинности является исчерпывающим описанием функции как математического объекта (хотя и возможно не раскрывает суть логической связки содержательно).

Таблица истинности — это таблица, которая показывает все возможные значения логических переменных и результат логической функции для каждой из этих комбинаций. Она помогает визуализировать и анализировать поведение логических функций.

Каждая строка таблицы истинности соответствует одной из возможных комбинаций значений входных переменных. Для двух переменных (A и B) будет $2^2 = 4$ строки. Для трех переменных - $2^3 = 8$, и так далее.

Задание: построить таблицу истинности для своего варианта.

1б. Построение СДНФ и СКНФ

Совершенная ДНФ — это способ представления логической функции, при котором она записывается как дизъюнкция (логическое «ИЛИ») конъюнкций (логическое «И») всех переменных (именно это означает «совершенность»). Каждая конъюнкция соответствует строке таблицы истинности, где функция принимает значение 1. Переменные, равне 0, инвертируются (через «НЕ»), а переменные, равные 1, записываются без изменений.

СДНФ позволяет представлять любую логическую функцию в максимально развернутом и упорядоченном виде. Она используется для точного описания работы логической схемы, так как каждая строка таблицы истинности, где функция равна 1, представляется соответствующим логическим выражением — сама функция с «уникальным» набором называется **минтермом** и устанавливает 1 лишь в одном, особом наборе значений логических переменных, который она, собственно, и характеризует. Таким образом, по умолчанию любой конъюнкт СДНФ возвращает 0, но если среди них найдется подходящий, то он выведет всю функцию в 1 по дизъюнкции.

СДНФ важна при разработке цифровых схем, чтобы реализовать конкретное поведение функции на всех возможных комбинациях входных переменных. СДНФ полезна в ситуациях, когда необходимо точно определить, при каких комбинациях входных сигналов схема должна выдавать логическую 1. Например, при проектировании цифровых устройств, таких как дешифраторы, мультиплексоры или автоматические системы управления, СДНФ помогает формализовать работу устройства.

СКНФ — это представление логической функции в виде конъюнкции (логическое «И») нескольких дизъюнкций (логическое «ИЛИ»). Каждая дизъюнкция соответствует строке таблицы истинности, где функция принимает значение 0. Переменные, равные 1, инвертируются (через «НЕ»), а переменные, равные 0, записываются без изменений. Таким образом получается **макстерм** — функция, возвращающая единицу по умолчанию, кроме ровно одного случая, за которым она и применяется в СКНФ.

СКНФ используется для обратного случая — когда нужно описать, при каких комбинациях переменных функция принимает значение 0. Это позволяет минимизировать и упорядочить

работу схем, особенно когда необходимо определить, при каких условиях схема будет выдавать логическую 0.

СКНФ активно используется при минимизации логических схем и упрощении вычислений. Она часто применяется в случае, когда важно знать, при каких условиях схема «отключена» (выдает 0), например, в логических схемах защиты, где требуется определить условия сбоя.

Таким образом, СДНФ «конструирует» функцию из единичек среди нулей, а СКНФ «выкалывает» нули из массы единичек. При помощи СДНФ невозможно построить константу 0, а при помощи СКНФ невозможно построить константу 1, в то время как СДНФ на 1(a,b,c,d, …) это перечисление всех возможных вариантов, как и для СКНФ будет 0(a,b,c,d, …).

Постройте совершенные конъюнктивные и дизъюнктивные нормальные формы Вашего варианта, соберите меньшую из них в схему, в симуляторе логических схем, например, simulator.io, logic.ly, Logisim и др., и проверьте корректность ее работы, протестировав схему на таблице истинности

1в. Минимизация до МДНФ и МКНФ посредством карты Карно

Физическая реализация булевской функции в форме СДНФ или СКНФ универсальна, но громоздка и не экономична как с точки зрения читаемости, так и с точки зрения затрат элементной базы (чем больше элементов задействовано, тем дороже и потенциально ненадежнее схема).

Задача минимизации сводится к наиболее экономичному представлению функции в заданном базисе. Традиционным является булев «базис», состоящий из отрицания НЕ, конъюнкции И и дизъюнкции ИЛИ. Причем для нормальных форм обязательным является отрицание выражений не более сложных, чем отдельно взятая переменная — для НФ нельзя отрицать какую-то часть выражения, только переменные, и с таким ограничением на отрицание-дополнение базис Буля и правда становится базисом в строго математическом смысле этого слова (как минимальный достаточный набор функций для построения произвольного выражения).

Для минимизации используют методы Куайна — МакКласки или карту Карно и другие.

(см. ПР7 на СДО)

1г. Построение минимальной схемы на элементах И-НЕ

Для изготовления схем используется иная элементная база, нежели чем для их разработки. Конкретно, достаточно часто применяются функции И-НЕ (NAND, отрицание конъюнкции) и ИЛИ-НЕ (NOR, отрицание дизъюнкции) — одна из двух на выбор.

Не останавливаясь на различиях между ними (а они есть, для интересующихся есть практический пример — ищем «nor flash vs nand flash»), с точки зрения логики в основе преобразований МДНФ и МКНФ в И-НЕ или ИЛИ-НЕ формы лежат законы Де Моргана:

- $(xy)' \equiv x' \vee y'$,
- $(x \vee y)' \equiv x'y'$.

Следствием из законов де Моргана является возможность переобозначить конъюнкцию через отрицание и дизъюнкцию, а дизъюнкцию — через отрицание и конъюнкцию:

- $xy \equiv (x' \vee y')'$

- $x \vee y \equiv (x'y')'$

Задача: переведите МДНФ или МКНФ на выбор по закону де Моргана, а затем соберите схему.

Задание «Программируемая логика». (перемещено в работу номер 6)

Программируемая логика достаточно широко используется в специализированном оборудовании в промышленности для автоматического управления сложными высокоответственными (промышленность) или высокопроизводительными (сетевое оборудование) аппаратно-программными системами. Этой темы мы коснемся несколько позже, поскольку она требует более детального рассмотрения.

Задание 2. Применение булевских связок в моделировании бизнес-процессов

Диаграмма в нотации BPMN предназначена для описания определенных сценариев и их развития в реальном мире с позиции ответственности и обработки различных вариантов развития событий.

Различают несколько видов шлюзов (Gateways), в частности: 1) работающие по XOR (исключающему ИЛИ), 2) параллельный шлюз И (обозначающий параллельно текущие процессы), 3) либо процессы могут течь по логике шлюза OR (несоисключающему ИЛИ, обыкновенной дизъюнкции):

Исключающий шлюз (эксклюзивный, Exclusive Gateway), «или/или» — выбор только одного пути;

Параллельный шлюз (Parallel Gateway), «и» — выбор всех путей;

Неисключающий шлюз (неэксклюзивный, Inclusive Gateway), «и/или» — выбор одного или нескольких;

Ярким примером XOR-шлюза является тот факт, что, зайдя в магазин с разделением входа и выхода (например, крупный супермаркет), у вас будет альтернатива внутри сценариев что делать после выбора товаров: выйти через него через кассы самообслуживания, или же через кассы с кассиром, и занять очередь в обе Вы не можете.

Для AND-шлюза примером может быть кооперативная подготовка двух студентов к зачету по билетам, где один выполняет одну часть работы, а другой — другую: без результатов

труда любого из них окончательный результат неполный, и следует ждать завершения всех подпроцессов.

Примером процесса с задачами, распределенными по OR-шлюзу, может стать коммуникация со знакомым через различные каналы связи: мы можем попробовать мессенджеры, телефон, СМС-сообщение и электронное письмо, и допускаются все варианты или любое подмножество из данных (главное не переборщить и не напугать адресата своей назойливостью).

Задача: описать некий бизнес-процесс (сценарий), в котором будут задействованы все три различные связки. Допускается одна дорожка (но можно больше если это требуется), обязательны начало и конец (концы сценария), а также наличие задач между любыми связками точно так, как это показывается в примерах (https://bpmn2.ru/blog/vse-shluzi-v-bpmn-s-primerami/).

Рекомендуется выполнять на сайте https://bpmn.io, либо с помощью Camunda Modeler (https://camunda.com/download/modeler/) или аналогичных (Bizagi Modeler).