Глава 14: Математические основы

Из SWEBOK

ВВЕДЕНИЕ

Профессионалы-программисты живут программами. На очень простом языке можно запрограммировать только то, что следует хорошо понятной недвусмысленной логике. Область знаний Mathematical Foundations (KA) помогает инженерам-программистам понять эту логику, которая, в свою очередь, переводится в код языка программирования. Математика, которой уделяется основное внимание в этом KA, сильно отличается от типичной арифметики, где рассматриваются и обсуждаются числа. Логика и рассуждения — это сущность математики, которой должен заниматься инженер-программист.

Математика, в некотором смысле, является изучением формальных систем. Слово «формальный» ассоциируется с точностью, поэтому двусмысленного или ошибочного толкования факта быть не может. Таким образом, математика — это изучение любых и всех определенных истин о любом понятии. Эта концепция может относиться как к числам, так и к символам, изображениям, звукам, видео — почти ко всему. Короче говоря, не только числа и числовые уравнения подлежат точности. Напротив, инженер-программист должен иметь точную абстракцию в разнообразной области приложения.

Содержание

Руководство *SWEBOK*Математические основы KA охватывает основные методы определения набора правил рассуждений в контексте изучаемой системы. Все, что можно вывести, следуя этим правилам, является абсолютной уверенностью в контексте этой системы. В этом KA определяются и обсуждаются методы, которые могут представлять и продвигать рассуждения и суждения инженера-программиста точным (и, следовательно, математическим) образом. Язык и методы логики, которые здесь обсуждаются, позволяют нам описывать математические доказательства, позволяющие сделать окончательный вывод об абсолютной истинности определенных понятий, выходящих за рамки чисел. Короче говоря, вы можете написать программу для решения задачи, только если она следует некоторой логике. Цель этого KA — помочь вам развить умение определять и описывать такую логику.

РАЗБИВКА ТЕМ ПО МАТЕМАТИЧЕСКИМ ОСНОВАМ

Разбивка тем для Mathematical Foundations KA показана на рисунке 14.1.

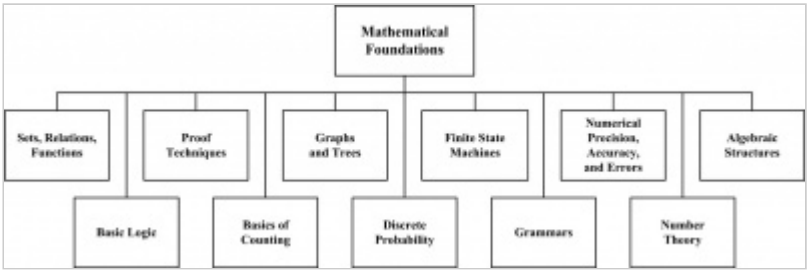

Рисунок 14.1: Разбивка тем для математических основ KA

1 набор, отношения, функции

[1 , c2]

Установить . Множество — это совокупность объектов, называемых элементами множества. Множество можно представить, перечислив его элементы между фигурными скобками, например, S = {1, 2, 3}.

Символ ∈ используется для обозначения того, что элемент принадлежит множеству или, другими словами, является членом множества. Его отрицание представлено ∉, например, 1 ∈ S, но 4 ∉ S.

В более компактном представлении набора с использованием нотации построителя наборов {x | P(x)} — это множество всех x, таких что P(x) для любого предложения P(x) в любом дискурсивном универсуме. Примеры некоторых важных наборов включают следующее:

- N = {0, 1, 2, 3, …} = множество неотрицательных целых чисел.
- Z = {…, −3, −2, −1, 0, 1, 2, 3, …} = множество целых чисел.

Конечное и бесконечное множество . Множество с конечным числом элементов называется конечным множеством. И наоборот, любое множество, не имеющее конечного числа элементов, является *бесконечным множеством* . Множество всех натуральных чисел, например, является бесконечным множеством.

Мощность . Мощность конечного множества S — это количество элементов в S. Это представлено |S|, например, если S = {1, 2, 3}, то |S| = 3.

Универсальный набор . В общем случае S = {x ∈ U | p(x)}, где U — это универсум дискурса, в котором должен интерпретироваться предикат P(x). «Вселенная дискурса» для данного предиката часто называется универсальным множеством. С другой стороны, универсальный набор можно определить как набор всех элементов.

Установить равенство . Два множества равны тогда и только тогда, когда они состоят из одних и тех же элементов, т. е.:

- X = Y ≡ ∀p (p ∈ X ↔ p ∈ Y).

Подмножество . X является подмножеством множества Y или X содержится в Y, если все элементы X входят в Y. Это обозначается X ⊆ Y. Другими словами, X ⊆ Y тогда и только тогда, когда ∀p (p ∈ X → p ∈ Y). Например, если X = {1, 2, 3} и Y = {1, 2, 3, 4, 5}, то X ⊆ Y. Если X не является подмножеством Y, оно обозначается как X ⊈ Y.

Правильное подмножество . X является собственным подмножеством Y (обозначается X ⊂ Y), если X является подмножеством Y, но не равно Y, т. е. в Y есть некоторый элемент, которого нет в X. Другими словами, X ⊂ Y, если (X ⊆ Y) ∧ (X ≠ Y). Например, если X = {1, 2, 3}, Y = {1, 2, 3, 4} и Z = {1, 2, 3}, то X ⊂ Y, но X не является правильным подмножеством Z. Множества X и Z равны. Если X не является собственным подмножеством Y, оно обозначается как X ⊄ Y.

Суперсет . Если X является подмножеством Y, то Y называется надмножеством X. Это обозначается Y ⊇ X, т. е. Y ⊇ X тогда и только тогда, когда X ⊆ Y. Например, если X = {1, 2, 3 } и Y = {1, 2, 3,4, 5}, то Y ⊇ X.

Пустой набор . Множество без элементов называется *пустым множеством* . Пустой набор, обозначаемый ∅, также называется нулевым или пустым набором.

Набор мощности . Набор всех подмножеств набора X называется *набором мощности* X. Он представлен как $\wp$ (X). Например, если X = {a, b, c}, то $\wp$(X) = {∅, {a}, {b}, {c}, {a, b}, {a, c}, {b, в}, {a, б, в}}. Если |X| = n, то $|\wp(X)| = 2^{n}$.

Диаграммы Венна . Диаграммы Венна — это графическое представление множеств в виде замкнутых областей на плоскости. Например, на рис. 14.2 прямоугольник представляет универсальное множество, а заштрихованная область — множество X.

1.1 Операции установки

Пересечение . Пересечение двух множеств X и Y, обозначаемое X ∩ Y, представляет собой множество общих элементов как в X, так и в Y. Другими словами, X ∩ Y = {p | (p ∈ X) ∧ (p ∈ Y)}. Как, например, {1, 2, 3} ∩ {3, 4, 6} = {3}. Если X ∩ Y = f, то два множества X и Y называются парой непересекающихся множеств.

Диаграмма Венна для пересечения множеств показана на рис. 14.3. Общая часть двух наборов представляет собой пересечение наборов.

Союз . Объединение двух множеств X и Y, обозначаемое X ∪ Y, представляет собой множество всех элементов либо в X, либо в Y, либо в обоих. Другими словами, X ∪ Y = {p | (p ∈ X) ∨ (p ∈ Y)}. Как, например, {1, 2, 3} ∪ {3, 4, 6} = {1, 2, 3, 4, 6}. Можно заметить, что |X ∪ Y| = |X| + |Y| − |X ∩ Y|.

Диаграмма Венна, иллюстрирующая объединение двух множеств, представлена заштрихованной областью на рис. 14.4.

Дополнение . Множество элементов универсального множества, не принадлежащих данному множеству X, называется его дополнительным множеством X'. Другими словами, X' = {p | (p ∈ U) ∧ (p ∉ X)}. Заштрихованная часть диаграммы Венна на рис. 14.5 представляет дополнительный набор X

Установить разность или относительное дополнение . Набор элементов, принадлежащих множеству X, но не множеству Y, образует отличие множества Y от X. Это представлено как X − Y. Другими словами, X − Y = {p | (p ∈ X) ∧ (p ∉ Y)}. Как, например, {1, 2, 3} - {3, 4, 6} = {1, 2}. Можно доказать, что X − Y = X ∩ Y'. Установленная разница X – Y показана заштрихованной областью на рис. 14.6 с использованием диаграммы Венна.

Декартово произведение . Обычная пара {p, q} — это множество из двух элементов. В наборе порядок элементов не имеет значения, поэтому {p, q} = {q, p}. В упорядоченной паре (p, q) важен порядок вхождения элементов. Таким образом, (p, q) ≠ (q, p), если только p = q. В общем случае (p, q) = (s, t) тогда и только тогда, когда p = s и q = t. Для двух множеств X и Y их декартово произведение X × Y — это множество всех упорядоченных пар (p, q), таких что p ∈ X и q ∈ Y. Другими словами, X × Y = {(p, q) | (p ∈ X) ∧ (q ∈ Y)}. Например, {a, b} × {1, 2} = {(a, 1), (a, 2), (б, 1), (б, 2)}

1.2 Свойства набора

Некоторые из важных свойств и законов множеств упомянуты ниже.

1. Ассоциативные законы:

X ∪ (Y ∪ Z) = (X ∪ Y) ∪ Z

X ∩ (Y ∩ Z) = (X ∩ Y) ∩ Z

2. Коммутативные законы:

X ∪ Y = Y ∪ X

X ∩ Y = Y ∩ X

3. Распределительные законы:

X ∪ (Y ∩ Z) = (X ∪ Y) ∩ (X ∪ Z)

X ∩ (Y ∪ Z) = (X ∩ Y) ∪ (X ∩ Z)

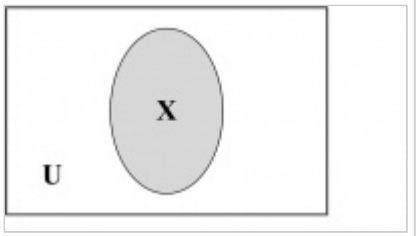

Рисунок 14.2: Диаграмма Венна для набора X

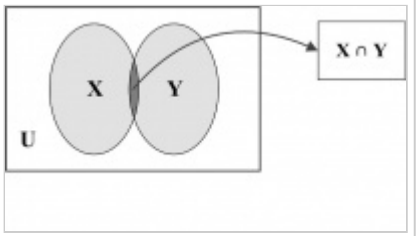

Рисунок 14.3: Пересечение множеств X и Y

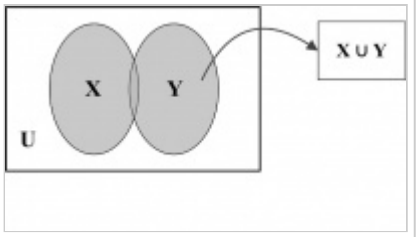

Рисунок 14.4: Объединение множеств X и Y

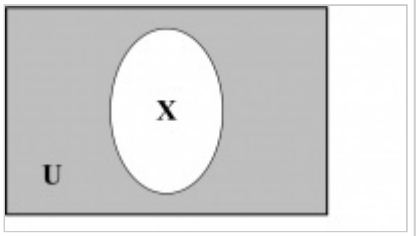

Рисунок 14.5: Диаграмма Венна для дополнительного множества X

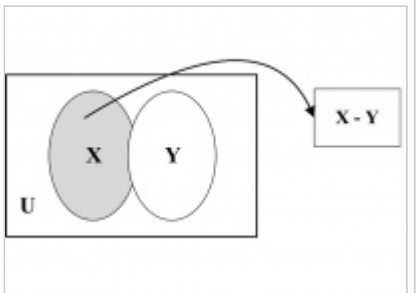

Рисунок 14.6: Диаграмма Венна для X − Y

4. Законы тождества:

$X \cup \emptyset = X$

$X \cap U = X$

5. Законы дополнений:

$X \cup X' = U$

$X \cap X' = \emptyset$

6. Идемпотентные законы:

$X \cup X = X$

$X \cap X = X$

7. Связанные законы:

$X \cup U = U$

$X \cap \emptyset = \emptyset$

8. Законы поглощения:

$X \cup (X \cap Y) = X$

$X \cap (X \cup Y) = X$

9. Законы Де Моргана:

$(X \cup Y)' = X' \cap Y'$

$(X \cap Y)' = X' \cup Y'$

1.3 Связь и функция

Отношение — это ассоциация между двумя наборами информации. Например, рассмотрим набор жителей города и их телефонных номеров. Сочетание имен с соответствующими телефонными номерами является отношением. Это сопряжение *упорядочено* для всего отношения. В рассматриваемом примере для каждой пары либо сначала идет имя, а затем номер телефона, либо наоборот. Набор, из которого извлекается первый элемент, называется *набором доменов,* а другой набор называется *набором диапазонов* . Домен — это то, с чего вы начинаете, а диапазон — это то, чем вы заканчиваете.

Функция — это *правильное* отношение. Отношение R(X, Y) хорошо себя ведет, если функция сопоставляет каждый элемент набора доменов X с одним элементом набора диапазонов Y. Давайте рассмотрим набор доменов X как набор людей, и пусть набор диапазонов Y хранит их телефоны. числа. Предполагая, что у человека может быть более одного телефонного номера, рассматриваемое отношение не является функцией. Однако если мы проведем связь между именами жителей и их датами рождения с именем, установленным в качестве домена, то это станет отношением хорошего поведения и, следовательно, функцией. Это означает, что хотя все функции являются отношениями, не все отношения являются функциями. В случае функции, заданной x, можно получить один и ровно один y для каждой упорядоченной пары (x, y).

Например, рассмотрим следующие два отношения.

- А: {(3, –9), (5, 8), (7, –6), (3, 9), (6, 3)}.
- Б: {(5, 8), (7, 8), (3, 8), (6, 8)}.

Это тоже функции? В случае отношения A доменом являются все значения x, т. е. {3, 5, 6, 7}, а диапазоном являются все значения y, т. е. {–9, –6, 3, 8, 9}. Отношение A не является функцией, так как есть два разных значения диапазона, –9 и 9, для одного и того же значения x, равного 3. В случае

отношения B домен такой же, как и для A, т. е. {3, 5 , 6, 7}. Однако диапазон представляет собой один элемент {8}. Это квалифицируется как пример функции, даже если все значения x сопоставляются с одним и тем же значением y. Здесь каждое значение x отличается, и, следовательно, функция работает хорошо. Отношение B может быть представлено уравнением y = 8.

Характеристика функции может быть проверена с помощью теста вертикальной линии, который указан ниже:

Если на графике отношения можно провести вертикальную линию, пересекающую график более чем в одном месте, то отношение не является функцией .

В этом примере обе линии L1 и L2 пересекают график отношения трижды. Это означает, что для одного и того же значения x существует три разных значения y для каждого случая. Таким образом, отношение не является функцией

2 Базовая логика

[1 , c1]

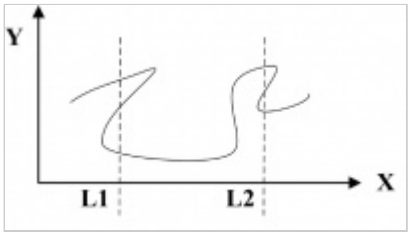

Рисунок 14.7: Тест вертикальной линии на функциональность

2.1 Пропозициональная логика

Предложение – это высказывание, которое либо истинно, либо ложно, но не то и другое одновременно. Рассмотрим повествовательные предложения, которым имеет смысл присвоить одно из двух значений статуса: *true или false* . Некоторые примеры предложений приведены ниже.

- 1. Солнце – звезда
- 2. Слоны — млекопитающие.
- 3. 2 + 3 = 5.

Однако a + 3 = b не является предложением, поскольку оно ни истинно, ни ложно. Это зависит от значений переменных *a* и *b* .

Закон исключенного третьего : для каждого предложения p либо истинно, либо ложно.

Закон противоречия : для каждого утверждения p не может быть, чтобы p было одновременно истинным и ложным.

Логика высказываний — это область логики, которая имеет дело с предложениями. Таблица истинности отображает отношения между значениями истинности предложений.

Логическая переменная — это переменная, значение которой либо истинно, либо ложно. Битовые операции компьютера соответствуют логическим операциям булевых переменных.

Необходимо изучить основные логические операторы, включая отрицание ($\neg$ p), конъюнкцию (p $\wedge$ q), дизъюнкцию (p $\vee$ q), исключающее ИЛИ (p $\oplus$ q) и импликацию (p $\rightarrow$ q). Составные предложения могут быть образованы с помощью различных логических операторов.

Составное предложение, которое всегда истинно, является тавтологией. Составное предложение, которое всегда ложно, есть противоречие. Составное предложение, которое не является ни тавтологией, ни противоречием, является случайностью.

Сложные предложения, которые всегда имеют одно и то же истинностное значение, называются логически эквивалентными (обозначаются $\equiv$). Некоторые из общих эквивалентов:

Законы тождества:

p $\wedge$ T $\equiv$ p

p $\vee$ F $\equiv$ p

Законы господства:

$p \vee T \equiv T$

$p \wedge F \equiv F$

Идемпотентные законы:

$п \vee п \equiv п$

$п \wedge п \equiv п$

Закон двойного отрицания:

$\neg (\neg p) \equiv p$

Коммутативные законы:

$п \vee д \equiv д \vee p$

$п \wedge д \equiv д \wedge p$

Ассоциативные законы:

$(p \vee q) \vee r \equiv p \vee (q \vee r)$

$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$

Распределительные законы:

$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$

Законы де Моргана:

$\neg (p \wedge q) \equiv \neg p \vee \neg q$

$\neg (p \vee q) \equiv \neg p \wedge \neg q$

2.2 Логика предикатов

Предикат — это шаблон глагольной фразы, который описывает свойство объектов или отношение между объектами, представленными переменными. Например, в предложении *Цветок красный* шаблон *красный* является сказуемым. Он описывает свойство цветка. Тот же сказуемый может использоваться и в других предложениях.

Предикатам часто дается имя, например, «Красный» или просто «R» может использоваться для представления предиката *красный* . Предполагая, что R в качестве имени для предиката - *red* , предложения, в которых утверждается, что объект имеет красный цвет, могут быть представлены как *R(x)* , где *x* представляет произвольный объект. *R(x)* читается как *x красный*

Квантификаторы позволяют делать утверждения о целых коллекциях объектов вместо того, чтобы перечислять объекты по именам.

Квантор универсальности ∀x утверждает, что предложение истинно для всех значений переменной x. Например, ∀x Tiger(x) → Mammal(x) означает, что все тигры являются млекопитающими.

Квантор существования ∃x утверждает, что предложение истинно хотя бы для одного значения переменной x. Например, ∃x Тигр(x) → Людоед(x) означает, что существует по крайней мере один тигр-людоед.

Таким образом, в то время как универсальная квантификация использует импликацию, экзистенциальная квантификация естественным образом использует конъюнкцию.

Переменная x, вводимая в логическое выражение квантором, связывается с ближайшим охватывающим квантором. Переменная называется свободной, если она не связана с квантором.

Точно так же в языке программирования с блочной структурой переменная в логическом выражении ссылается на ближайший квантификатор, в области действия которого она появляется.

Например, в ∃x (Cat(x) ∧ ∀x (Black(x))) x в Black(x) универсально определяется количественно. Выражение подразумевает, что кошки существуют и все черное.

Логика высказываний не может представить многие утверждения, используемые в информатике и математике. Он также не может сравнить эквивалентность и некоторые другие типы отношений между предложениями.

Например, утверждение a *больше 1* не является предложением, потому что невозможно сделать вывод, истинно оно или ложно, не зная значения *a* . Таким образом, логика высказываний не может иметь дело с такими предложениями. Однако такие утверждения довольно часто встречаются в математике, и мы хотим сделать вывод на основе этих утверждений. Кроме того, модель, связанная со следующими двумя логическими эквивалентностями, не может быть захвачена логикой высказываний: *«Не все мужчины курят»* и *«Некоторые мужчины не курят».*Каждое из этих двух предложений рассматривается независимо в логике высказываний. В логике высказываний нет механизма, позволяющего выяснить, эквивалентны ли они друг другу. Следовательно, в логике высказываний каждое эквивалентное предложение рассматривается индивидуально, а не имеет дело с общей формулой, которая охватывает все эквивалентности вместе.

Предполагается, что логика предикатов является более мощной логикой, которая решает эти проблемы. В некотором смысле логика предикатов (также известная как логика первого порядка или исчисление предикатов) является расширением логики высказываний на формулы, включающие термины и предикаты.

3 метода доказательства

[1 , c1]

Доказательство — это аргумент, строго устанавливающий истинность утверждения. Сами доказательства могут быть представлены формально в виде дискретных структур.

Утверждения, используемые в доказательстве, включают аксиомы и постулаты, которые по существу являются основополагающими предположениями о математических структурах, гипотезами доказываемой теоремы и ранее доказанными теоремами.

Теорема – это утверждение, истинность которого можно доказать.

Лемма — это простая теорема, используемая при доказательстве других теорем.

Следствие — это предложение, которое может быть установлено непосредственно из доказанной теоремы.

Гипотеза – это утверждение, истинность которого неизвестна.

Когда доказательство гипотезы найдено, гипотеза становится теоремой. Много раз догадки оказываются ложными и, следовательно, не являются теоремами.

3.1 Методы доказательства теорем

Прямое доказательство . Прямое доказательство - это метод, позволяющий установить, что импликация $p \rightarrow q$ верна, путем демонстрации того, что q должно быть истинным, когда истинно p.

Например, чтобы показать, что если n нечетно, то $n^2 - 1$ четно, предположим, что n нечетно, т. е. $n = 2k + 1$ для некоторого целого числа k: $\therefore n^2 = (2k + 1)^2 = 4k^2 + 4k + 1$

Поскольку первые два члена правой части (RHS) являются четными числами независимо от значения k, левая часть (LHS) (т. е. n^2) является нечетным числом. Следовательно, $n^2 - 1$ четно.

Доказательство от противного . Предложение p истинно от противного, если оно доказано на основе истинности импликации $\neg p \rightarrow q$, где q — противоречие.

Например, чтобы показать, что сумма 2x + 1 и 2y - 1 четна, предположим, что сумма 2x + 1 и 2y - 1 нечетна. Другими словами, 2(x + y), кратное 2, нечетно. Это противоречие. Следовательно, сумма 2x + 1 и 2y − 1 четна.

Правило вывода — это образец, устанавливающий, что если все посылки верны, то можно сделать вывод, что определенное утверждение вывода истинно. Необходимо изучить основные правила сложения, упрощения и конъюнкции.

Доказательство по индукции . Доказательство по индукции проводится в два этапа. Во-первых, устанавливается, что предложение верно для базового случая — обычно для положительного целого числа 1. На втором этапе устанавливается, что если предложение верно для произвольного положительного целого числа k, то оно также должно выполняться для следующего большего числа. целое число, k + 1. Другими словами, доказательство по индукции основано на правиле вывода, согласно которому истинность бесконечной последовательности утверждений P(n), $\forall n \in [1 \ldots \infty]$, устанавливается, если P(1) верно, а во-вторых, $\forall k \in [2 \ldots n]$, если $P(k) \rightarrow P(k + 1)$.

Здесь можно отметить, что для доказательства по математической индукции не предполагается, что P(k) истинно для всех положительных целых чисел k. Доказательство теоремы или предложения требует от нас только установить, что если предполагается, что P(k) истинно для любого произвольного натурального числа k, то P(k + 1) также истинно. Корректность математической индукции как надежного метода доказательства выходит за рамки обсуждения в данном тексте. Докажем по индукции следующее предложение.

Предложение: сумма первых n положительных нечетных целых чисел P(n) равна n^2.

Базовый шаг: Предложение верно для n = 1, поскольку $P(1) = 1^2 = 1$. Базовый шаг завершен.

Индуктивный шаг: Гипотеза индукции (IH) состоит в том, что предложение верно для n = k, где k — произвольное положительное целое число k.

$\therefore 1 + 3 + 5+ \ldots + (2k - 1) = k^2$

Теперь нужно показать, что $P(k) \rightarrow P(k + 1)$.

$P(k + 1) = 1 + 3 + 5+ \ldots +(2k - 1) + (2k + 1) = P(k) + (2k + 1) = k^2 + (2k + 1)$ [с использованием IH] = $к^2 + 2к + 1 = (к + 1)^2$

Таким образом, показано, что если предложение верно при n = k, то оно верно и при n = k + 1.

Базисный шаг вместе с индуктивным шагом доказательства показывают, что P (1) истинно, а условное утверждение $P (k) \rightarrow P (k + 1)$ истинно для всех натуральных k. Следовательно, предложение доказано.

4 Основы счета

[1 , с6]

Правило сумм гласит, что если задачу t_1 можно выполнить n_1 способами, а вторую задачу t_2 можно выполнить n_2 способами, и если эти задачи нельзя выполнить одновременно, то существует $n_1 + n_2$ способа выполнить любую задачу.

- Если A и B непересекающиеся множества, то $|A \cup B|=|A| + |Б|$.

- В общем, если A1, A2, …. , An — непересекающиеся множества, то |A1 ∪ A2 ∪ … ∪ An| = |A1| + |A2| + … + |Ан|.

Например, если 200 спортсменов участвуют в спринтерских соревнованиях и 30 спортсменов участвуют в прыжках в длину, то сколько существует способов выбрать одного спортсмена, который является либо спринтером, либо прыгуном в длину? Используя правило сумм, ответ будет 200 + 30 = 230.

Правило произведения гласит, что если задачу t_1 можно выполнить n_1 способами, а вторую задачу t_2 можно выполнить n_2 способами после того, как первая задача была выполнена, то существует $n_1 * n_2$ способов выполнить первое задание. процедура.

- Если A и B непересекающиеся множества, то |A × B| = |A| * |Б|.
- В общем случае, если A1, A2, …, An — непересекающиеся множества, то |A1 × A2 × … × An| = |A1| * |A2| * …. * |Ан|.

Например, если 200 спортсменов участвуют в спринтерских соревнованиях и 30 спортсменов участвуют в прыжках в длину, то сколько существует способов выбрать двух спортсменов, чтобы один был спринтером, а другой прыгал в длину? Используя правило произведения, ответ будет 200 * 30 = 6000.

Принцип *включения-исключения* гласит, что если задачу t_1 можно выполнить n_1 способами, а вторую задачу t_2 можно выполнить n_2 способами одновременно с t_1, то для нахождения общего числа способов можно выполнить две задачи, вычесть количество способов выполнить обе задачи из $n_1 + n_2$.

- Если A и B не пересекаются, |A ∪ B| = |A| + |Б| − |A ∩ B|.

Другими словами, принцип включения-исключения направлен на то, чтобы объекты на пересечении двух наборов не учитывались более одного раза.

Рекурсия — это общий термин для практики определения объекта с точки зрения самого себя. Существуют рекурсивные алгоритмы, рекурсивно определенные функции, отношения, множества и т. д.

Рекурсивная функция — это функция, которая вызывает сама себя. Например, мы определяем f(n) = 3 * f(n − 1) для всех n ∈ N, n ≠ 0 и f(0) = 5.

Алгоритм является рекурсивным, если он решает проблему, сводя ее к экземпляру той же проблемы с меньшими входными данными.

Явление называется случайным, если отдельные результаты неопределенны, но долгосрочная картина многих отдельных результатов предсказуема.

Вероятность любого исхода для случайного явления — это доля случаев, когда результат будет иметь место в очень длинной серии повторений.

Вероятность P(A) любого события A удовлетворяет условию 0 ≤ P(A) ≤ 1. Любая вероятность — это число от 0 до 1. Если S — выборочное пространство в вероятностной модели, то P(S) = 1. Все возможные исходы вместе должны иметь вероятность 1.

Два события A и B называются непересекающимися, если они не имеют общих исходов и поэтому никогда не могут произойти вместе. Если A и B — два непересекающихся события, то P(A или B) = P(A) + P(B). Это известно как правило сложения для непересекающихся событий.

Если два события не имеют общих исходов, вероятность того, что одно или другое произойдет, равна сумме их индивидуальных вероятностей.

Перестановка — это расположение объектов, в котором порядок имеет значение без повторения. Можно выбрать r объектов в определенном порядке из n объектов, используя $^{n}P_{r}$ способов, где $^{n}p_{r}$ = n! / (n - r)!. Различные обозначения, такие как $^{n}P_{r}$ и P (n, r), используются для представления количества перестановок набора из n объектов, взятых по r за раз.

Комбинация — это выбор объектов, в которых порядок не имеет значения без повторения. Это отличается от перестановки, потому что порядок не имеет значения. Если меняется только порядок (но не члены), то новая комбинация не образуется. Можно выбрать r объектов в любом порядке из n объектов, используя $^{n}C_{r}$ способов, где $^{n}C_{r}$ = n! / [p! * (n − r)!].

5 Графики и деревья

[1 , c10, c11]

5.1 Графики

Граф G = (V, E), где V — множество вершин (узлов), а E — множество ребер. Ребра также называются дугами или звеньями.

F — функция, отображающая набор ребер E в набор упорядоченных или неупорядоченных пар элементов V. Например, на рис. 14.8 G = (V, E), где V = {A, B, C}, E = {e1, e2, e3} и F = {(e1, (A,C)), (e2, (C, B)), (e3, (B, A))}.

Граф на рис. 14.8 — это простой граф, состоящий из набора вершин или узлов и набора ребер, соединяющих неупорядоченные пары. Ребра в простых графах неориентированы. Такие графы также называются неориентированными графами. Например, на рис. 14.8 (e1, (A, C)) можно заменить на (e1, (C, A)), так как пара между вершинами A и C неупорядочена. Это верно и для двух других ребер. В мультиграфе одни и те же две вершины могут соединяться более чем одним ребром. Два или более соединительных ребра между одной и той же парой вершин могут отражать множественные связи между одними и теми же двумя вершинами. Такие ребра называются параллельными или кратными ребрами.

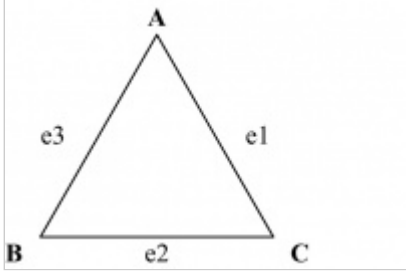

Рисунок 14.8: Пример графика

Например, на рис. 14.9 ребра e3 и e4 находятся между A и B. Рис. 14.9 представляет собой мультиграф, в котором ребра e3 и e4 являются кратными ребрами.

В *псевдографе* разрешены ребра, соединяющие узел сам с собой. Такие ребра называются петлями.

Например, на рис. 14.10 ребро e4 начинается и заканчивается в B. Рис. 14.10 — это псевдограф, в котором e4 — петля.

Ориентированный граф G = (V, E) состоит из множества вершин V и множества ребер E, являющихся упорядоченными парами элементов V. Ориентированный граф может содержать петли.

Например, на рис. 14.11 G = (V, E), где V = {A, B, C}, E = {e1, e2, e3} и F = {(e1, (A, C)), (e2, (B, C)), (e3, (B, A))}.

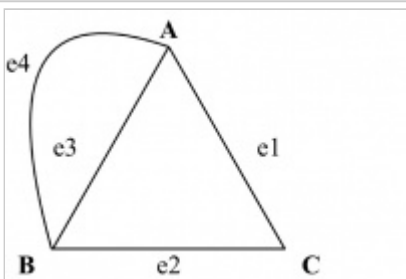

Рисунок 14.9: Пример мультиграфа

Во взвешенном графе G = (V, E) каждое ребро имеет связанный с ним вес. Вес ребра обычно представляет собой числовое значение, связанное с отношениями между соответствующими двумя вершинами.

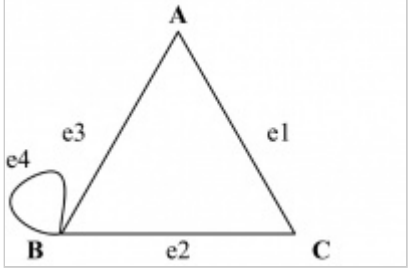

Рисунок 14.10: Пример псевдографа

Например, на рис. 14.12 веса ребер e1, e2 и e3 приняты равными 76, 93 и 15 соответственно. Если вершины A, B и C представляют три города в штате, весами могут быть, например, расстояния в милях между этими городами.

Пусть G = (V, E) — неориентированный граф с множеством ребер E. Тогда для ребра e ∈ E, где e = {u,v}, часто используются следующие термины:

- u, v называются *смежными , соседними* или связанными .
- ребро e *инцидентно* вершинам u и v.
- ребро e *соединяет* u и v.
- вершины u и v являются *конечными точками* ребра e.

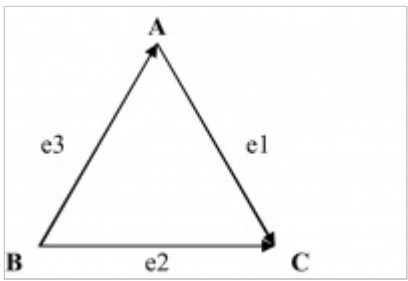

Рисунок 14.11: Пример ориентированного графа

Если вершина v ∈ V, множество вершин неориентированного графа G(V, E), то:

- степень *v* , deg(v), есть количество инцидентных ребер, за исключением того, что любые петли учитываются дважды.
- вершина со степенью 0 называется *изолированной вершиной* .
- вершина степени 1 называется *висячей вершиной* .

Пусть G(V, E) — ориентированный граф. Если e(u, v) является ребром G, то часто используются следующие термины:

- u *смежно с* v, а v *смежно с* u.
- e *происходит от* u и *переходит* в v.
- e *соединяет* u с v или e *идет от* u к v.
- начальная *вершина* e есть u
- конечная *вершина* e есть v.

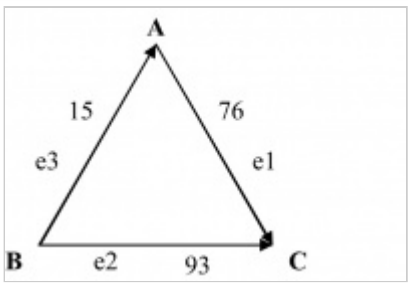

Рисунок 14.12: Пример взвешенного графика

Если вершина v принадлежит множеству вершин ориентированного графа G(V, E), то

- *in-степень* v, deg−(v) — это количество ребер, идущих к v, т. е. для которых v — конечная вершина.
- *исходящая степень* v, deg+(v), — это количество ребер, выходящих из v, т. е. таких, для которых v — начальная вершина.
- *степень* v, $deg(v) = deg^{-}(v) + deg^{+}(v)$, представляет собой сумму vs степени входа и степени выхода.
- петля в вершине вносит 1 как в степень входа, так и в степень выхода этой вершины.

Можно отметить, что, следуя приведенным выше определениям, степень узла неизменна независимо от того, считаем ли мы его ребра направленными или ненаправленными.

В неориентированном графе путь длины n из u в v — это последовательность n смежных ребер из вершины u в вершину v.

- Путь является *цепью,* если u=v.
- Путь *пересекает* вершины вдоль него.
- Путь называется *простым* , если он не содержит ни одного ребра более одного раза.

Цикл на n вершинах C_n для любого $n \geq 3$ — это простой граф, где $V = \{v_1, v_2, \ldots, v_n\}$ и $E = \{\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}\}$.

Например, на рис. 14.13 показаны два цикла длиной 3 и 4.

Список смежности — это таблица с одной строкой на вершину, в которой перечислены смежные с ней вершины. Список смежности для ориентированного графа поддерживает список конечных узлов для каждой вершины в графе.

Например, на рис. 14.14 показаны списки смежности для псевдографа на рис. 14.10 и ориентированного графа на рис. 14.11. Поскольку исходящая степень вершины C на рис. 14.11 равна нулю, в списке смежности нет записи относительно вершины C.

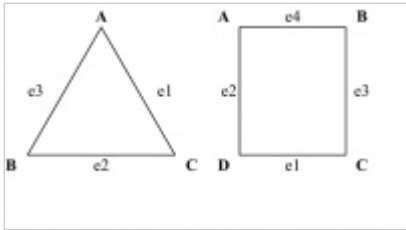

Рисунок 14.13: Пример циклов C_3 и C_4

Необходимо изучить различные представления графа, такие как матрица смежности, матрица инцидентности и списки смежности.

Vertex	Adjacency List
A	B, C
B	A, B, C
C	A, B

Рисунок 14.9: Списки смежности для графов на рисунках 14.10 и 14.11

5.2 Деревья

Дерево T(N, E) — это иерархическая структура данных n = |N| узлов со специально назначенным корневым узлом R, а оставшиеся n − 1 узлов образуют поддеревья под корневым узлом R. Количество ребер |E| в дереве всегда будет равно |N| − 1.

Поддерево в узле X — это подграф дерева, состоящий из узла X и его потомков, а также всех ребер, инцидентных этим потомкам. В качестве альтернативы этому рекурсивному определению дерево может быть определено как связный неориентированный граф без простых схем.

Однако следует помнить, что дерево по своей природе строго иерархично по сравнению с графом, который является плоским. В случае дерева упорядоченная пара строится между двумя узлами как родительский и дочерний. Каждый дочерний узел в дереве связан только с одним родительским узлом, тогда как это ограничение становится бессмысленным для графа, где не существует связи родитель-потомок.

Неориентированный граф является деревом тогда и только тогда, когда существует единственный простой путь между любыми двумя его вершинами.

На рис. 14.15 представлено дерево T(N, E), в котором множество узлов N = {A, B, C, D, E, F, G, H, I, J, K}. Множество ребер E равно {(A, B), (A, C), (A, D), (B, E), (B, F), (B, G), (C, H), (C , I), (D, J), (D, K)}.

Родителем некорневого узла v является уникальный узел u с направленным ребром от u к v. У каждого узла в дереве есть уникальный родительский узел, кроме корня дерева.

Например, на рис. 14.15 корневой узел A является родительским узлом для узлов B, C и D. Точно так же B является родительским узлом для E, F, G и т. д. Корневой узел A не имеет родителя.

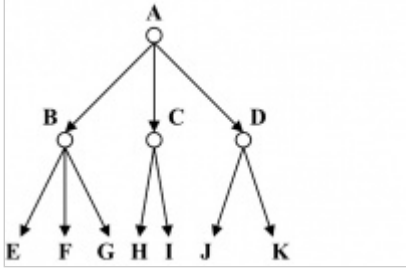

Рисунок 14.15: Пример дерева

Узел, у которого есть потомки, называется внутренним узлом. Например, на рис. 14.15 узел A или узел B являются примерами внутренних узлов. Степень узла в дереве равна количеству его потомков. Например, на рис. 14.15 корневой узел A и его дочерний узел B имеют степень 3. Узлы C и D имеют степень 2.

Расстояние узла от корневого узла с точки зрения количества прыжков называется его *уровнем* . Узлы в дереве находятся на разных уровнях. Корневой узел находится на уровне 0. С другой стороны, уровень узла X — это длина уникального пути от корня дерева до узла X.

Например, корневой узел A находится на уровне 0 на рис. 14.15. Узлы B, C и D находятся на уровне 1. Все остальные узлы на рис. 14.15 находятся на уровне 2. Высота дерева — это максимальное количество уровней узлов в дереве. Например, на рис. 14.15 высота дерева равна 2.

Узел называется *листом* , если он не имеет потомков. Степень листового узла равна 0. Например, на рис. 14.15 все узлы от E до K являются листовыми узлами со степенью 0. Предками или предшественниками некорневого узла X являются все узлы на пути от корня к узлу X. Например, на рис. 14.15 узлы A и D образуют множество предков для J.

Преемниками или потомками узла X являются все узлы, у которых X является предком. Для дерева с n узлами все оставшиеся n - 1 узлы являются преемниками корневого узла. Например, на рис. 14.15 узел B имеет преемников в E, F и G. *Если узел X является предком узла Y, то узел Y является преемником узла* X.

Два или более узлов, совместно использующих один и тот же родительский узел, называются *родственными* узлами. Например, на рис. 14.15 узлы E и G являются одноуровневыми. Однако узлы E и J, хотя и относятся к одному уровню, не являются родственными узлами. *Два одноуровневых узла имеют один и тот же уровень, но два узла на одном уровне не обязательно являются одноуровневыми* .

Дерево называется *упорядоченным деревом* , если относительное положение вхождений дочерних узлов существенно. Например, генеалогическое древо является упорядоченным деревом, если, как правило, имя старшего брата или сестры стоит всегда впереди (т. е. слева от) младшего брата или сестры.

В неупорядоченном дереве относительное положение вхождений между братьями и сестрами не имеет никакого значения и может быть изменено произвольно.

Двоичное дерево формируется с нулем или более узлов, где есть корневой узел R, а все остальные узлы образуют пару упорядоченных поддеревьев под корневым узлом.

В бинарном дереве ни один внутренний узел не может иметь более двух дочерних элементов. Однако надо учитывать, что кроме этого критерия по степени внутренних узлов бинарное дерево всегда упорядочено. Если позиции левого и правого поддеревьев для любого узла в дереве меняются местами, то получается новое дерево. Например, на рис. 14.16 два бинарных дерева различны, поскольку позиции вхождений дочерних элементов A различны в двух деревьях.

Согласно [1*], бинарное дерево называется полным бинарным деревом, если каждый внутренний узел имеет ровно двух потомков. Например, бинарное дерево на рис. 14.17 является полным бинарным деревом, так как оба внутренних узла A и B имеют степень 2. Полное бинарное дерево, соответствующее приведенному выше определению, также называется *строго бинарным деревом* .

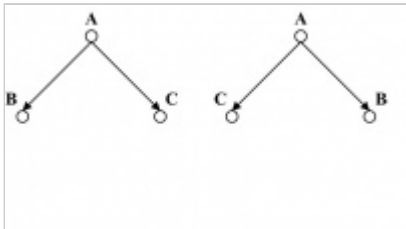

Рисунок 14.16: Примеры бинарных деревьев

Например, оба бинарных дерева на рис. 14.18 являются полными бинарными деревьями. Дерево на рис. 14.18(a) является как полным, так и полным бинарным деревом. В полном бинарном дереве все уровни, за исключением, возможно, последнего, заполнены до отказа. В случае, если последний уровень полного бинарного дерева не заполнен, узлы появляются с крайних левых доступных позиций.

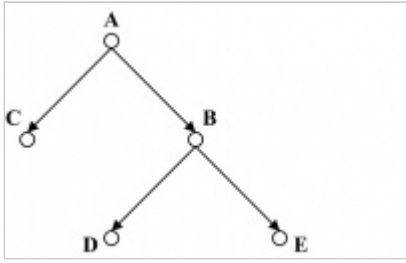

Рисунок 14.17: Пример полного бинарного дерева

Интересно, что согласно приведенным выше определениям дерево на рис. 14.18(b) является полным, но не полным бинарным деревом, поскольку узел B имеет только одного дочернего элемента в D. Напротив, дерево на рис. 14.17 является полным, но не полным, бинарное дерево, так как дочерние элементы B встречаются в дереве, а дочерние элементы C не появляются на последнем уровне. Двоичное дерево высоты H является сбалансированным, если все его листовые узлы находятся на уровнях H или H − 1. Например, все три бинарных дерева на рисунках 14.17 и 14.18 являются сбалансированными бинарными деревьями.

В бинарном дереве высоты H не более 2^H листьев. Другими словами, если бинарное дерево с L листьями полное и сбалансированное, то его высота $H = \lceil \log_2 L \rceil$.

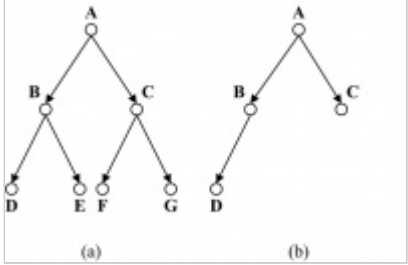

Рисунок 14.18: Пример полных бинарных деревьев

Например, это утверждение верно для двух деревьев на рисунках 14.17 и 14.18(a), поскольку оба дерева полные и сбалансированные. Однако приведенное выше выражение не соответствует дереву на рис. 14.18(b), поскольку оно не является полным бинарным деревом.

Бинарное дерево поиска (BST) — это особый вид двоичного дерева, в котором каждый узел содержит отдельное значение ключа, а значение ключа каждого узла в дереве меньше, чем каждое значение ключа в его правом поддереве, и больше, чем каждое значение ключа. в его левом поддереве.

Алгоритм обхода — это процедура систематического посещения каждого узла бинарного дерева. Обходы дерева могут быть определены рекурсивно.

Если T является двоичным деревом с корнем R, а оставшиеся узлы образуют упорядоченную пару ненулевого левого поддерева T_L и непустого правого поддерева T_R ниже R, то функция обхода в прямом порядке PreOrder (T) определяется как:

Предзаказ(T) = R, Предзаказ($TL_)$, Предзаказ(T_R)… ур. 1

Рекурсивный процесс поиска обхода поддеревьев в прямом порядке продолжается до тех пор, пока поддеревья не будут признаны нулевыми. Здесь запятые используются в качестве разделителей для удобства чтения.

Постпорядок и порядок могут быть определены аналогичным образом с использованием уравнения. 2 и ур. 3 соответственно.

PostOrder(T) = PostOrder($TL_)$, PostOrder(T_R), R … eqn 2

InOrder(T) = InOrder(T_L), R, InOrder(T_R) …уравнение 3

Например, дерево на рис. 14.19 — это бинарное дерево поиска (BST). Предварительный порядок, обратный порядок и обход по порядку для BST приведены ниже в соответствующем порядке.

Выход предзаказа: 9, 5, 2, 1, 4, 7, 6, 8, 13, 11, 10, 15

Вывод по почте: 1, 4, 2, 6, 8, 7, 5, 10, 11, 15, 13, 9

Вывод по порядку: 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 13, 15

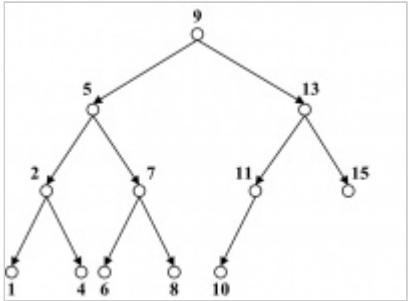

Рисунок 14.19: Двоичное дерево поиска

Дальнейшее обсуждение деревьев и их использования было включено в раздел 6 «Структура и представление данных» книги Computing Foundations KA.

6 Дискретная вероятность

[1 , c7]

Вероятность — это математическое описание случайности. Основное определение вероятности и случайности дано в разделе 4 настоящего ЗК. Здесь давайте начнем с концепций распределения вероятностей и дискретной вероятности.

Вероятностная модель — это математическое описание случайного явления, состоящее из двух частей: выборочного пространства S и способа присвоения вероятностей событиям. Демонстрационное пространство определяет множество всех возможных исходов, тогда как событие — это подмножество

выборочного пространства, представляющее возможный исход или набор исходов.

Случайная величина — это функция или правило, которое присваивает номер каждому результату. По сути, это просто символ, представляющий результат эксперимента.

Например, пусть X будет количеством орла, когда эксперимент подбрасывает монету n раз. Точно так же пусть S будет скоростью автомобиля, зарегистрированной радар-детектором.

Значения случайной величины могут быть дискретными или непрерывными в зависимости от эксперимента. Дискретная случайная величина может содержать все возможные результаты, не пропуская ни одного, хотя это может занять бесконечное количество времени. Непрерывная случайная величина используется для измерения бесчисленного количества значений, даже если дано бесконечное количество времени.

Например, если случайная величина X представляет результат, являющийся действительным числом от 1 до 100, то X может иметь бесконечное количество значений. Никогда нельзя перечислить все возможные исходы для X, даже если отведено бесконечное количество времени. Здесь X — непрерывная случайная величина. Наоборот, для того же интервала от 1 до 100 можно использовать другую случайную переменную Y для перечисления всех целочисленных значений в диапазоне. Здесь Y — дискретная случайная величина.

Заглавная буква, например X, будет представлять *имя* случайной величины. Его аналог в нижнем регистре x будет представлять *значение* случайной величины.

Вероятность того, что случайная величина X будет равна x, равна:

P(X = x) или, проще говоря, P(x).

Функция распределения (плотности) вероятностей — это таблица, формула или график, описывающие значения случайной величины и вероятность, связанную с этими значениями.

Вероятности, связанные с дискретными случайными величинами, обладают следующими свойствами:

я. $0 \leq P(x) \leq 1$ для всех x

II. $\Sigma P(x) = 1$

Дискретное распределение вероятностей можно представить как дискретную случайную величину.

Среднее значение μ модели распределения вероятностей представляет собой сумму условий произведения для отдельных событий и вероятности их исхода. Другими словами, для возможных исходов $x_1, x_2, \ldots, x_n$ в выборочном пространстве S, если p_k – вероятность исхода x_k, среднее значение этой вероятности будет равно $\mu = x_1 p_1 + x_2 п_2 + \ldots + Икс_п п_п$.

X	1	2	3	4	5	6
P(x)	1/6	1/6	1/6	1/6	1/6	1/6

Рисунок 14.20: Дискретная функция вероятности для катящегося кубика

Например, среднее значение плотности вероятности для распределения на рис. 14.20 будет равно

$1 * (1/6) + 2 * (1/6) + 3 * (1/6) + 4 * (1/6) + 5 * (1/6) + 6 * (1/6) = 21 * (1/6) = 3,5$

Здесь выборочное пространство относится к набору всех возможных результатов.

Дисперсия s^2 дискретной вероятностной модели равна: $s^2 = (x_1 - \mu)^2 p_1 + (x_2 - \mu)^2 p_2 + \ldots + (x_k - \mu)^2 p_k$.

Стандартные *отклонения* представляют собой квадратный корень из дисперсии.

Например, для распределения вероятностей на рис. 14.20 вариация σ^2 будет равна

$c^2 = [(1 – 3,5)^2 * (1/6) + (2 – 3,5)^2 * (1/6) + (3 – 3,5)^2 * (1/6) + (4 – 3,5)^2 * (1/6) + (5 – 3,5)^2 * (1/6) + (6 – 3,5)^2 * (1/6)] = (6,25 + 2,25 + 0,25 + 0,5 + 2,25 + 6,25) * (1/ 6) = 17,5 * (1/6) = 2,90$

∴ стандартное отклонение s =

Эти числа действительно направлены на получение среднего значения из повторных экспериментов. Это основано на единственном наиболее важном явлении вероятности, т. е. на том, что среднее значение из повторных экспериментов, вероятно, будет близко к ожидаемому значению одного эксперимента. Более того, среднее значение, скорее всего, будет ближе к ожидаемому значению любого одного эксперимента по мере увеличения количества экспериментов.

7 конечных автоматов

[1 , c13]

Компьютерная система может быть абстрагирована как отображение из состояния в состояние, управляемое входными данными. Другими словами, систему можно рассматривать как функцию перехода T: S × I → S × O, где S — множество состояний, а I, O — входная и выходная функции. Если множество состояний S конечно (не бесконечно), система называется *конечным автоматом* (FSM).

С другой стороны, конечный автомат (FSM) представляет собой математическую абстракцию, состоящую из конечного числа состояний и переходов между этими состояниями. Если область S × I достаточно мала, то можно указать T явно, используя диаграммы, подобные блок-схеме, чтобы проиллюстрировать, как работает логика для различных входных данных. Однако это практично только для машин с очень небольшой информационной емкостью.

FSM имеет конечную внутреннюю память, функцию ввода, которая считывает символы последовательно и по одному, и функцию вывода.

Работа автомата начинается с начального состояния, проходит через переходы в зависимости от ввода в разные состояния и может заканчиваться в любом допустимом состоянии. Однако лишь некоторые из всех состояний отмечают успешный ход операции. Это называется *состояниями принятия* .

Информационная емкость автомата C = log |S|. Таким образом, если мы представим машину с информационной емкостью C бит в виде конечного автомата, то ее граф переходов состояний будет иметь |S| = 2 узла ^{C}.

Конечный автомат формально определяется как M = (S, I, O, f, g, s_0).

- S — множество состояний;
- I — набор входных символов;
- O — набор выходных символов;
- f — функция перехода состояний;
- g — выходная функция;
- s_0 — начальное состояние.

При наличии входных данных x ∈ I в состоянии S_k автомат переходит в состояние $_{Sh}$ в соответствии с функцией перехода состояний f и выдает результат y ∈ O с использованием выходной функции g.

Например, на рис. 14.21 показан FSM с S0 $_в$ качестве начального состояния и S1 $_в$ качестве конечного состояния. Здесь S = {S0 $_{, S1}$, S2 $_}$; я = {0, 1}; O = {2, 3}; f(S0 , 0) = S2 , f(S0 $_,$ 1) = S1 $_,$ f(S1 $_{, 0}$) = S2 $_,$ f(S1 , $_1$) = S2 $_,$ f($_{S2}$, $_0$) = S_2 , f(S_2 , 1) = S_0 ; g(S0 , 0) = 3, g(S0 $_,$ 1) = 2, g(S1 , $_0$) = 3, g(S1 $_{, 1}$) =

2, $g(S_2, 0) = 2$, $g(S2, 1) = 3$.

Переход состояний и выходные значения для разных входов в разных состояниях могут быть представлены с использованием таблицы состояний. Таблица состояний для FSM на рис. 14.21 показана на рис. 14.22. Каждая пара напротив входного символа представляет новое состояние и выходной символ.

Например, рисунки 14.22(a) и 14.22(b) представляют собой два альтернативных представления конечного автомата на рисунке 14.21.

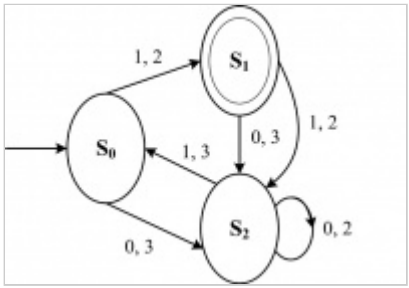

Рисунок 14.21: Пример конечного автомата

8 грамматик

[1 , c13]

Грамматика естественного языка говорит нам, составляет ли комбинация слов правильное предложение. В отличие от естественных языков, формальный язык определяется четко определенным набором правил синтаксиса. Допустимые предложения формального языка могут быть описаны грамматикой с помощью этих правил, называемых *продукционными* .

Current State	Input 0	Input 1
S_0	S_2	S_1
S_1	S_2	S_2
S_2	S_2	S_0

(a)

Current State	Output Input 0	Output Input 1	State Input 0	State Input 1
S_0	3	2	S_2	S_1
S_1	3	2	S_2	S_2
S_2	2	3	S_2	S_0

(b)

Рисунок 14.22: Табличное представление конечного автомата

Формальный язык — это набор слов или строк конечной длины в некотором конечном алфавите, а грамматика определяет правила формирования этих слов или строк. Весь набор слов, допустимых для грамматики, составляет язык грамматики. Таким образом, грамматика G — это любое компактное точное математическое определение языка L, а не просто необработанный список всех допустимых предложений языка или примеров этих предложений.

Грамматика подразумевает алгоритм, который будет генерировать все допустимые предложения языка. Существуют разные типы грамматик.

Фраза-структура или грамматика типа 0 G = (V, T, S, P) представляет собой 4-кортеж, в котором:

- V — словарный запас, т. е. набор слов.
- $T \subseteq V$ — множество слов, называемых терминалами.
- $S \in N$ — специальное слово, называемое начальным символом.
- P — множество продукционных правил замены одного фрагмента предложения другим.

Существует другое множество слов $N = V - T$, называемое нетерминалом. Нетерминалы представляют такие понятия, как *существительное* . Производственные правила применяются к строкам, содержащим нетерминалы, до тех пор, пока в строке не останется нетерминальных символов. Начальный символ S является нетерминалом.

Язык, порожденный формальной грамматикой G, обозначаемый L(G), представляет собой множество всех строк в множестве алфавитов V, которые можно сгенерировать, начиная с начального символа, применяя продукционные правила до тех пор, пока не будут заменены все нетерминальные символы. в строке.

Например, пусть G = ({S, A, a, b}, {a, b}, S, {S → aA, S → b, A → aa}). Здесь множество терминалов N = {S, A}, где S — начальный символ. Три продукционных правила для грамматики задаются как P1: S → aA; P2: C → б; P3: A → aa.

Применяя продукционные правила всеми возможными способами, из начального символа можно сгенерировать следующие слова.

S → aA (с использованием P1 на начальном символе)

→ aaa (с использованием P3)

S → b (с использованием P2 на начальном символе)

Ничего другого для G вывести нельзя. Таким образом, язык грамматики G состоит всего из двух слов: L(G) = {aaa, b}.

8.1 Распознавание языка

Формальные грамматики можно классифицировать в соответствии с допустимыми типами продукции. Иерархия Хомского (введенная Ноамом Хомским в 1956 г.) описывает такую схему классификации.

Как показано на рис. 14.23, мы делаем следующие выводы о различных типах грамматик:

- 1. Каждая регулярная грамматика является контекстно-свободной грамматикой (КСГ).
- 2. Каждая CFG является контекстно-зависимой грамматикой (CSG).
- 3. Каждая CSG является грамматикой фразовой структуры (PSG).

Контекстно-зависимая грамматика : все фрагменты в правой части либо длиннее соответствующих фрагментов в левой части, либо пусты, т. е. если b → a, то |b| < |a| или a = ∅.

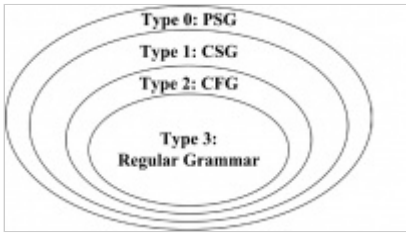

Рисунок 14.23: Иерархия грамматик Хомского

Формальный язык является контекстно-зависимым, если его порождает контекстно-зависимая грамматика. Контекстно-свободная грамматика: все фрагменты в LHS имеют длину 1, т. е. если A → a, то |A| = 1 для всех A ∈ N.

Термин контекстно-свободный происходит от того факта, что A всегда можно заменить на a, независимо от контекста, в котором оно встречается.

Формальный язык является контекстно-свободным, если его порождает контекстно-свободная грамматика. Контекстно-свободные языки являются теоретической основой синтаксиса большинства языков программирования.

Регулярная грамматика . Все фрагменты в RHS являются либо одиночными терминалами, либо парой, построенной терминалом и нетерминалом; т. е. если A → a, то либо a ∈ T, либо a = cD, либо a = Dc для c ∈ T, D ∈ N.

Если a = cD, то грамматика называется линейной справа грамматикой. С другой стороны, если a = Dc, то грамматика называется леволинейной грамматикой. И праволинейные, и леволинейные грамматики являются регулярными или грамматиками типа 3.

Язык L(G), порожденный регулярной грамматикой G, называется регулярным языком.

Регулярное выражение A — это строка (или шаблон), образованная из следующих шести элементов информации: a ∈ S, набор алфавитов, e, 0 и операции ИЛИ (+), ПРОИЗВЕДЕНИЕ (.), КОНКАТЕНАЦИЯ (*) . Язык G, L(G) равен всем тем строкам, которые соответствуют G, L(G) = {x ∈ S*|x соответствует G}.

Для любого a ∈ S L(a) = a; L(e) = {ε}; L(0) = 0. + функционирует как или, L(A + B) = L(A) ∪ L(B). . создает структуру продукта, L(AB) = L(A) . ФУНТ). обозначает конкатенацию, L(A*) = {x1x2…xn | xi ∈ L(A) и n ³ 0}

Например, регулярному выражению (ab)* соответствует набор строк: {e, ab, abab, ababab, abababab, …}.

Например, регулярное выражение (aa)* соответствует набору строк, состоящих из одной буквы a и имеющих четную длину.

Например, регулярное выражение (aaa)* + (aaaaa)* соответствует набору строк длины, кратной 3 или 5.

9 Числовая точность, точность и ошибки

[2 c21

[2 , 62]

Основной целью численного анализа является разработка эффективных алгоритмов вычисления точных числовых значений функций, решений алгебраических и дифференциальных уравнений, оптимизационных задач и т. д.

Дело в том, что все цифровые компьютеры могут хранить только конечные числа. Другими словами, компьютер никак не может представить бесконечно большое число — будь то целое число, рациональное число, любое действительное или все комплексные числа (см. раздел 10 «Теория чисел»). Таким образом, математика приближения становится очень важной для обработки всех чисел в конечном диапазоне, который может обрабатывать компьютер.

Каждому числу в компьютере назначается место или слово, состоящее из определенного количества двоичных цифр или битов. K битовое слово может хранить в общей сложности $N = 2^k$ различных чисел.

Например, компьютер, использующий 32-битную арифметику, может хранить в общей сложности $N = 2^{32} \approx 4{,}3 \times 10^9$ различных чисел, тогда как другой, использующий 64 бита, может обрабатывать $N' = 2^{64} \approx 1{,}84 \times 10^{19}$ различных чисел. Вопрос в том, как распределить эти N чисел по реальной линии для максимальной эффективности и точности практических вычислений.

Один очевидный выбор — распределить их равномерно, что приведет к арифметике с фиксированной точкой. В этой системе первый бит в слове используется для представления знака, а остальные биты обрабатываются как целые значения. Это позволяет представлять целые числа от $1 - ½N$, т. е. от $= 1 - 2^{k-1}$ до 1. В качестве аппроксимирующего метода это не годится для нецелых чисел.

Другой вариант состоит в том, чтобы расположить числа близко друг к другу, скажем, с равномерным промежутком 2^{-n}, и таким образом равномерно распределить общее количество N чисел по интервалу $-2^{-n-1} < x \leq 2^{-n-1}$ N. Действительные числа лежащие между промежутками, представлены либо *округлением* (что означает ближайший точный представитель), либо *разрезанием* (что означает точный представитель непосредственно под — или выше, если отрицательное — число).

Числа, лежащие за пределами диапазона, должны быть представлены наибольшим (или наибольшим отрицательным) числом, которое может быть представлено. Это становится символом переполнения. Переполнение происходит, когда вычисление дает значение, превышающее максимальное значение в диапазоне.

Когда скорость обработки является существенным узким местом, использование представлений с фиксированной запятой является привлекательной и более быстрой альтернативой более громоздкой арифметике с плавающей запятой, наиболее часто используемой на практике.

Давайте определим пару очень важных терминов: *точность* и *прецизионность* , связанные с численным анализом.

Точность — это близость, с которой измеренное или вычисленное значение согласуется с истинным значением. С другой стороны, точность — это близость, с которой два или более измеренных или вычисленных значения для одного и того же физического вещества согласуются друг с другом. Другими словами, точность — это близость, с которой число представляет точное значение.

Пусть x будет действительным числом, а x* будет приближением. Абсолютная ошибка приближения $x^* \approx x$ определяется как $| x^* - x |$. Относительная ошибка определяется как отношение абсолютной ошибки к величине x, т. е. $|x^* - x| / | x |$, что предполагает $x \neq 0$; в противном случае относительная ошибка не определяется.

Например, 1000000 является приближением к 1000001 с абсолютной ошибкой 1 и относительной ошибкой 10^{-6}, а 10 является приближением к 11 с абсолютной ошибкой 1 и относительной ошибкой 0,1. Как правило, относительная ошибка более интуитивно понятна и является предпочтительным показателем

размера ошибки. Существующее соглашение состоит в том, что ошибки всегда ≥ 0 и равны 0 тогда и только тогда, когда аппроксимация точна.

Аппроксимация x* имеет k значащих десятичных цифр, если ее относительная ошибка $< 5 \times 10^{-k-1}$. Это означает, что первые k цифр числа x*, следующих за первой ненулевой цифрой, совпадают с цифрами x.

Значащие цифры — это цифры числа, о которых известно, что они правильные. В измерение включается одна неопределенная цифра.

Например, измерение длины линейкой 15,5 мм с максимально допустимой погрешностью ±0,5 мм имеет 2 значащих цифры, тогда как измерение той же длины штангенциркулем, записанное как 15,47 мм с максимально допустимой погрешностью ±0,01 мм, имеет 3 значащих цифры. .

10 Теория чисел

[1 , c4]

Теория чисел — одна из старейших ветвей чистой математики и одна из крупнейших. Конечно, это касается вопросов о числах, обычно означающих целые числа и дробные или рациональные числа. Различные типы чисел включают целые числа, действительные числа, натуральные числа, комплексные числа, рациональные числа и т. д.

10.1 Делимость

Начнем этот раздел с краткого описания каждого из вышеперечисленных типов чисел, начиная с натуральных чисел.

Натуральные числа . Эта группа чисел начинается с 1 и продолжается: 1, 2, 3, 4, 5 и так далее. Зеро не входит в эту группу. В группе натуральных чисел нет ни отрицательных, ни дробных чисел. Общим математическим символом для множества всех натуральных чисел является N.

Целые числа . В эту группу входят все натуральные числа плюс число 0.

К сожалению, не все принимают приведенные выше определения натуральных и целых чисел. Кажется, нет общего согласия о том, следует ли включать 0 в набор натуральных чисел.

Многие математики считают, что в Европе последовательность натуральных чисел традиционно начиналась с 1 (греки даже не считали числом 0). В 19 веке теоретики множеств и другие математики ввели соглашение о включении 0 в набор натуральных чисел.

Целые числа . В эту группу входят все целые числа и их отрицательные числа. Общим математическим символом для множества всех целых чисел является Z, т. е. $Z = \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$.

Рациональные числа . Это любые числа, которые можно выразить как отношение двух целых чисел. Общим символом для множества всех рациональных чисел является Q. Рациональные числа можно разделить на три типа в зависимости от того, как действуют десятичные дроби. Десятичные дроби либо не существуют, например, 15, либо, когда десятичные дроби существуют, они могут заканчиваться, как в 15.6, или повторяться по образцу, как в 1.666..., (что равно 5/3).

Иррациональные числа . Это числа, которые не могут быть выражены как целое число, деленное на целое число. Эти числа имеют десятичные дроби, которые никогда не заканчиваются и никогда не повторяются с шаблоном, например, PI или $\sqrt{2}$.

Реальные числа . Эта группа состоит из всех рациональных и иррациональных чисел. Числа, которые встречаются при изучении алгебры, являются действительными числами. Общий математический символ для множества всех действительных чисел — R.

Воображаемые числа . Все они основаны на воображаемом числе i . Это мнимое число равно квадратному корню из −1. Любое действительное число, кратное i , является мнимым числом, например, i , $5i$, $3.2i$, $-2.6i$ и т. д.

Комплексные числа . Комплексное число представляет собой комбинацию действительного числа и мнимого числа в форме a + b *i* . Действительная часть равна a, а b называется мнимой частью. Общий математический символ для множества всех комплексных чисел — C.

Например, 2 + 3 *i* , 3−5 *i* , 7,3 + 0 *i* и 0 + 5 *i* . Рассмотрим последние два примера:

7,3 + 0 *i* совпадает с действительным числом 7,3. Таким образом, все действительные числа являются комплексными числами с нулем в мнимой части. Точно так же 0 + 5 *i* — это просто мнимое число 5 *i* . Таким образом, все мнимые числа являются комплексными числами с нулем в действительной части.

Элементарная теория чисел предполагает делимость целых чисел. Пусть a, b ∈ Z, где a ≠ 0. Выражение a|b, т . е . *a, делит b* , если ∃c ∈ Z: b = ac, т. е. существует целое число c такое, что c, умноженное на a, равно b.

Например, 3|−12 верно, а 3|7 ложно.

Если *a* делит *b* , то мы говорим, что a является *множителем* b *или* a *является* делителем *b* , а *b* кратно *a* . *b* четно тогда и только тогда, когда 2| *б* .

Пусть *a* , *d* ∈ Z с d > 1. Тогда *mod d* обозначает остаток *r* от алгоритма деления с делимым *a* и делителем *d* , т. е. остаток от деления a на d. Мы можем вычислить (*мод d*) по формуле: *a* − *d* * ⌊a/d⌋, где ⌊a/d⌋ представляет пол действительного числа.

Пусть Z^{+} = {n ∈ Z | n > 0} и *a* , *b* ∈ Z, m ∈ Z+, то a конгруэнтно *b по модулю m* , записанному как *a* ≡ *b* (*mod m*), тогда и только тогда, когда *m* | *a-б* .

10.2 Простое число, НОД

Целое число p > 1 является простым тогда и только тогда, когда оно не является произведением любых двух целых чисел, больших 1, т. е. p является простым, если p > 1 ∧ ∃ ¬ a, b ∈ N: a > 1, b > 1, a * б = p.

Единственными положительными делителями простого числа p являются 1 и само число p. Например, числа 2, 13, 29, 61 и т. д. являются простыми числами. Непростые целые числа больше 1 называются составными числами. Составное число может быть составлено путем умножения двух целых чисел больше 1.

Есть много интересных применений простых чисел; среди них схема криптографии с открытым ключом, которая включает обмен открытыми ключами, содержащими произведение *p* * *q* двух случайных больших простых чисел *p* и *q* (закрытый ключ), который должен храниться в секрете данной стороной. Наибольший общий делитель gcd(a, b) целых чисел a, b — это наибольшее целое число d, являющееся делителем как числа a, так и числа b, т. е.

d = gcd(a, b) для max(d: d|a ∧ d|b)

Например, gcd(24, 36) = 12.

Целые числа *a* и *b* называются взаимно простыми или взаимно простыми тогда и только тогда, когда их НОД равен 1.

Например, ни 35, ни 6 не являются простыми, но они взаимно просты, поскольку эти два числа не имеют общих делителей больше 1, поэтому их НОД равен 1.

Набор целых чисел X = {i_1 , i_2 , …} взаимно прост, если все возможные пары i_h , i_k , h ≠ k, взятые из множества X, взаимно просты.

11 алгебраических структур

В этом разделе вводятся несколько представлений, используемых в высшей алгебре. Алгебраическая структура состоит из одного или двух множеств, замкнутых относительно некоторых операций и удовлетворяющих ряду аксиом, в том числе ни одной.

Например, группа, моноид, кольцо и решетка являются примерами алгебраических структур. Каждый из них определен в этом разделе.

11.1 Группа

Множество S, замкнутое относительно бинарной операции • образует группу, если бинарная операция удовлетворяет следующим четырем критериям:

- Ассоциативный: ∀a, b, c ∈ S, выполняется равенство (a • b) • c = a • (b • c).
- Тождество: существует такой элемент идентичности I ∈ S, что для всех a ∈ S выполняется равенство I • a = a • I = a.
- Обратный: каждый элемент a ∈ S имеет обратный элемент a' ∈ S по отношению к бинарной операции, т. е. a • a' = I; например, набор целых чисел Z относительно операции сложения является группой. Элемент идентичности набора равен 0 для

операция сложения. ∀x ∈ Z, обратным значением x будет –x, которое также входит в Z.

- Свойство замыкания: ∀a, b ∈ S, результат операции a • b ∈ S.
- Группа, которая является коммутативной, т. е. a • b = b • a, называется коммутативной или абелевой группой.

Множество натуральных чисел N (с операцией сложения) не является группой, так как для любого x > 0 в множестве натуральных чисел нет обратного. Таким образом, третье правило (обратного) для нашей операции нарушается. Однако множество натуральных чисел имеет некоторую структуру.

Множества с ассоциативной операцией (первое условие выше) называются полугруппами; если они также имеют элемент идентичности (второе условие), то они называются моноидами.

Таким образом, наш набор натуральных чисел при сложении является примером моноида, структуры, которая не совсем является группой, потому что в ней отсутствует требование, чтобы каждый элемент имел обратный по операции.

Моноид – это множество S, замкнутое относительно одной ассоциативной бинарной операции • и имеющее единичный элемент I ∈ S такой, что для всех a ∈ S выполняется I • a = a • I = a. Моноид должен содержать хотя бы один элемент.

Например, множество натуральных чисел N образует коммутативный моноид при сложении с единичным элементом 0. Тот же набор натуральных чисел N также образует моноид при умножении на единичный элемент 1. Множество натуральных чисел P образует коммутативный моноид при умножении с элементом идентичности 1.

Можно отметить, что, в отличие от элементов группы, элементы моноида могут не иметь обратных элементов. Моноид также можно рассматривать как полугруппу с единичным элементом.

Подгруппа — *это* группа H , содержащаяся внутри большей, G , такая, что единичный элемент G содержится в H , и всякий раз, когда h_1 и h_2 находятся в H , тогда также $h_1 \cdot h_2$ и h_1^{-1} . Таким образом, элементы H , снабженные групповой операцией на G , ограниченной H , действительно образуют группу.

Для любого подмножества S группы G подгруппа порождает S и их инверсии. Это наименьшая подгруппа G , содержащая S .

Например, пусть G — абелева группа, элементами которой являются $G' = \{0, 2, 4, 6, 1, 3, 5, 7\}$, *а* групповая операция — сложение по модулю 8. Эта группа имеет пару нетривиальных подгрупп: $J = \{0, 4\}$ и $H = \{0, 2, 4, 6\}$, где J также является подгруппой H .

В теории групп циклическая группа — это группа, которая может быть порождена одним элементом в том смысле, что в группе есть элемент a (называемый *генератором* группы), такой, что при мультипликативной записи каждый элемент группы равен мощность a . Группа G называется циклической, если $G = \{ a^n$ для любого целого $n \}$.

Поскольку любая группа, порожденная элементом в группе, является подгруппой этой группы, показать, что единственная подгруппа группы G, которая содержит a, является самой G, достаточно, чтобы показать, что G циклическая.

Например, группа $G = \{0, 2, 4, 6, 1, 3, 5, 7\}$ относительно операции сложения по модулю 8 является циклической. Подгруппы $J = \{0, 4\}$ и $H = \{0, 2, 4, 6\}$ также являются циклическими.

11.2 Кольца

Если мы возьмем абелеву группу и определим на ней вторую операцию, будет найдена новая структура, отличная от простой группы. Если эта вторая операция ассоциативна и дистрибутивна над первой, то мы имеем кольцо.

Кольцом называется тройка вида (S, +, •), где (S, +) — абелева группа, (S, •) — полугруппа, • дистрибутивна над +; т. е. «a, b, c ∈ S, справедливо равенство a • (b + c) = (a • b) + (a • c). Далее, если • коммутативно, то кольцо называется коммутативным. Если для операции • имеется элемент идентичности, то говорят, что кольцо имеет идентичность.

Например, (Z, +, *), т. е. множество целых чисел Z с обычными операциями сложения и умножения, является кольцом. Так как (Z, *) коммутативно, это кольцо является коммутативным или абелевым кольцом. Кольцо имеет 1 в качестве элемента идентичности.

Заметим, что вторая операция может не иметь элемента идентичности, и нам не нужно искать обратную операцию для каждого элемента по отношению к этой второй операции. Что касается того, что означает дистрибутив, то интуитивно это то, что мы делаем в элементарной математике, выполняя следующую замену: a * (b + c) = (a * b) + (a * c).

Поле — это кольцо, для которого элементы множества, кроме 0, образуют абелеву группу со второй операцией.

Простым примером поля является поле рациональных чисел (R, +, *) с обычными операциями сложения и умножения. Числа формата $a / b \in R$, где a , b — целые числа и $b \neq 0$. Аддитивная обратная дробь — это просто — a / b , а мультипликативная обратная — b / a при условии, что $a \neq 0$.

ИСПОЛЬЗОВАННАЯ ЛИТЕРАТУРА

[1] К. Розен, *Дискретная математика и ее приложения* , 7-е изд., McGraw-Hill, 2011.

[2] Э. У. Чейни и Д. Р. Кинкейд, *Численная математика и вычисления* , 6-е изд., Брукс/Коул, 2007.

БЛАГОДАРНОСТИ

Автор выражает благодарность профессору Аруну Кумару Чаттерджи, бывшему заведующему кафедрой математики Манипурского университета, Индия, и профессору Девадатте Синхе, бывшему заведующему кафедрой компьютерных наук и инженерии Калькуттского университета, Индия, в подготовка этой главы по математическим основам.