Кольца и поля

М.Д. Малых, РУДН

29 августа 2022 г.

Содержание

1. Определения

Определение 1. Кольцо (ring) — множество A, на котором заданы две бинарные операции, называемые сложение $(+)$ и умножение $(\cdot)$, со следующими свойствами, выполняющимися для любых $a, b, c \in A$:

1) $a + b = b + a$ (коммутативность сложения);

2) $a \cdot b = b \cdot a$ (коммутативность умножения);

3) $a + (b + c) = (a + b) + c$ (ассоциативность сложения);

4) $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ (ассоциативность умножения);

5) $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$ и $(b + c) \cdot a = (b \cdot a) + (c \cdot a)$ (дистрибутивность);

6) существует такой элемент $0 \in A$, что $a + 0 = 0 + a = a$ (существование нуля);

7) существует такой элемент $1 \in A$, что $a \cdot 1 = 1 \cdot a = a$ (существование единицы);

8) для любого $a \in A$ существует такой элемент $b \in A$, что $a+b = b+a = 0$ (существование противоположного элемента относительно сложения).

Замечание. Иногда понятие кольца трактуют шире, отказавшись от коммутативность умножения и существования единицы. Ниже под кольцом подразумевается коммутативное кольцо с единицей или явно оговорено противное.

Определение 2. Поле (field) — кольцо K, обладающее дополнительным свойством: для любого его элемента $a \neq 0$ существует такой элемент $b \in K$, что $ab = ba = 1$.

Базовые примеры:

1) Множество натуральных чисел $\mathbb{N}$ не является кольцом, поскольку не содержит 0.

2) Множество целых чисел $\mathbb{Z}$ является кольцом, но не является полем.

3) Множество рациональных чисел $\mathbb{Q}$ является полем.

4) Множество вещественных чисел $\mathbb{R}$ является полем.

Есть немало других свойств, которыми обладают все перечисленные множества. Большинство из них отсутствуют среди аксиом в определении 1 по той причине, что их можно вывести из указанных.

Пример 1. Докажем, что в кольце имеется только один нулевой элемент. Пусть A — кольцо, его элемент θ называют нулем, если для любого $a \in A$ верно $a + \theta = \theta + a = a$. К сожалению, верно и

$$0 + \theta = 0,$$

поскольку θ — нуль кольца, и

$$0 + \theta = \theta,$$

поскольку 0 — нуль кольца. Таким образом, все нули кольца равны друг другу. Это очень досадно, поскольку в противном случае можно было бы рассматривать нули как бесконечно малые числа и дать алгебраическое изложение математическому анализу.

2. Целые и рациональные числа в Sage

Sage — система компьютерной алгебры, в качестве внешнего языка используется Python, к которому добавлено множество новых функций и классов.

```
sage: NN
Non negative integer semiring
sage: ZZ
Integer Ring
sage: QQ
Rational Field
```

Функции `is_ring` и `is_field` позволяют проверить, являются ли множества кольцами или полями:

```
sage: ZZ.is_ring()
True
sage: ZZ.is_field()
False
sage: QQ.is_field()
True
```

Замечание. В Python функция может применяться слева и справа через точку, причем во втором случае ее называют методом:

```
sage: QQ.is_field()
```

```
True
sage: is_field(QQ)
True
```

Обычно результаты применения идентичны. Различия могут проявиться, если функция с одним именем определена несколько раз. Напр, если мы переопределим функцию `is_field`, то при вызове этой функции слева будет взята функция из пользовательских определений, а при ее вызове как метода из определения, данного при описании того класса, которому принадлежит $\mathbb{Q}$:

```
sage: def is_field(x): return 8
sage: is_field(QQ)
8
sage: QQ.is_field()
True
```

Целые и рациональные числа, равно как и действия с ними, вводятся обычным образом:

```
sage: 1+ 2/3
5/3
```

При этом можно выяснить, считает ли система то или иное выражение элементом того или иного кольца:

```
sage: 5 in QQ
True
sage: 5 in ZZ
True
sage: 5/2 in ZZ
False
sage: 5/2 + 8 in QQ
True
```

```
sage: sqrt(2) in QQ
False
```

При этом:

```
sage: type(5)
<class 'sage.rings.integer.Integer'>
sage: type(5/2)
<class 'sage.rings.rational.Rational'>
sage: type(sqrt(5))
<class 'sage.symbolic.expression.Expression'>
```

Таким образом, целые числа считаются по умолчанию элементами кольца целых чисел, рациональные — поля рациональных чисел, а выражения, которые система не может упростить до рационального числа, — символьными выражениями.

3. Вещественные числа в Sage

Поле вещественных чисел реализовано в Sage как множество десятичных дробей, для хранения которых требуется не более 53 бит:

```
sage: RR
Real Field with 53 bits of precision
```

Всякая десятичная дробь по умолчанию считается элементом $\mathbb{R}$:

```
sage: type(0.3)
<class 'sage.rings.real_mpfr.RealLiteral'>
```

Если система считает, что число принадлежит $\mathbb{R}$, то можно узнать соответствующую ему десятичную дробь:

```
sage: pi in RR
True
sage: RR(pi)
```

```
3.14159265358979
```

Однако здесь проявляется несколько тонких моментов.

1) С точки зрения синтаксиса вопрос `a in A` означает, может ли система рассматривать `a` как элемент множества `A`. Ответ False не означает, что `a` действительно не принадлежит `A`. Напр., путем значительных усилий[1] можно доказать равенство

$$2\arctan\frac{1}{2} - \arctan\frac{1}{7} = \frac{\pi}{4}.$$

Однако, Sage не знает этого равенства и поэтому не может преобразовать выражение

$$2\arctan\frac{1}{2} - \arctan\frac{1}{7} - \frac{\pi}{4}$$

в рациональное число, о чем и сообщает:

```
sage: 2*atan(1/2) - atan(1/7) - pi/4  in QQ
False
```

При этом:

```
sage: 2*atan(1/2) - atan(1/7) - pi/4  in RR
True
sage: RR(2*atan(1/2) - atan(1/7) - pi/4)
0.000000000000000
```

2) Если система может рассматривать `a` как элемент множества `A`, то `A(a)` — представление `a` как элемента множества `A`. Напр., $\frac{1}{3}$ является бесконечной десятичной дробью $0.333\ldots$. Система может рассмотреть это число как элемент `RR`, но его представление в этом множестве будет уже конечной десятичной дробью:

```
sage: 1/3 in RR
```

[1] C. Störmer. Solution complète en nombres entiers de l'équation $m\arctan 1/x + n\arctan 1/y = k\pi/4$. // Bulletin de la S. M. F., tome 27 (1899), p. 160-170. URL: `http://www.numdam.org/item?id=BSMF_1899__27__160_1`.

```
True
sage: RR(1/3)
0.333333333333333
```

Таким образом, при переходе к реализации поля $\mathbb{R}$ в Sage делается небольшая ошибка, которую называют ошибкой округления.

3) Реализация поля $\mathbb{R}$ в Sage не является полем, результат зависит от порядка слагаемых:

```
sage: RR(10^20)+RR(1) - RR(10^20)
0.000000000000000
sage: RR(10^20)- RR(10^20) +RR(1)
1.00000000000000
```

4. Символьные выражения

Если в процессе вычисления того или иного числа переходят к десятичным дробям, то делают ошибку округления и в итоге ответ не может быть точным. Такого рода манипуляция называют приближенными вычислениями (numerical calculation). Им противопоставляют символьные вычисления (symbolic computation). Термин этот очень трудно определить, однако сами символьные вычисления хорошо известны школьникам: именно они выполняются всякий раз, когда требуется упростить выражение. Главным признаком символьных вычислений и является отсутствие ошибки округления. Наука о символьных вычислениях на компьютере получила название компьютерной алгебры.

Сама идея «упрощения» состоит в том, что полученное в ходе решение той или иной задачи выражение нужно постараться записать в как можно более простом виде с тем, чтобы далее его было проще вычислить приближенно. Напр., пусть в ходе решение геометрической задачи получился ответ $\sqrt{2+\sqrt{9}}$. В школьном курсе его требуют упростить до $\sqrt{5}$, ведь в таком случае придется вычислять квадратный корень лишь один раз.

Здесь появилось несколько терминов, которые мы будем употреблять далее повсеместно: *символьное выражение* и *символьные вычисления*. Мы уклонимся от определения этих понятий, поскольку таковое неизбежно приведет к определению понятия символ.

В Sage знакомые со школы выражения и трактуются как символьные выражения (symbolic expression), элементы огромного кольца символьных выражений (SR, symbolic ring):

```
sage: SR
Symbolic Ring
sage: type(pi)
<class 'sage.symbolic.expression.Expression'>
sage: type(sin(1))
<class 'sage.symbolic.expression.Expression'>
sage: type(sqrt(2))
<class 'sage.symbolic.expression.Expression'>
```

На самом деле, это понятие трактуется даже шире, чем в школе. Напр., выражение, содержащее десятичные дроби, тоже трактуется как символьное выражение.

Помимо целых чисел и знаков, обозначающих те или иные операции с числами, выражения могут содержать буквы, трактуемые как символьные переменные (var, variables). В первых системах компьютерной алгебры всякая буква или комбинация букв, набранных без связывающих их арифметических действий, трактовалась как переменная. Однако в Sage требуется заранее указывать имена используемых символьных переменных.

```
sage: var("x,y")
(x, y)
sage: x^2+pi*y
x^2 + pi*y
```

```
sage: type(x^2+pi*y)
<class 'sage.symbolic.expression.Expression'>
```

Всякое символьное выражение можно постараться упростить. Простейшие упрощения делаются самой системой без каких либо предупреждений:

```
sage: sqrt(2+sqrt(9))
sqrt(5)
```

Более сложные делаются при помощи метода `simplify` и `full_simplify`:

```
sage: 2*sin(x)*cos(x)-sin(2*x)
2*cos(x)*sin(x) - sin(2*x)
sage: (2*sin(x)*cos(x)-sin(2*x)).simplify()
2*cos(x)*sin(x) - sin(2*x)
sage: (2*sin(x)*cos(x)-sin(2*x)).full_simplify()
0
```

Эти методы описаны только в описании класса SR и поэтому пишутся справа через точку. Они отличаются друг от друга набором правил, по которым выполняются упрощения.

Результат применения этих функций не всегда совпадает с ожиданиями пользователя. Система пытается получить более простое выражение, но у нее это не всегда получается. Напр., она далеко не всегда может понять, что выражение равно нулю.

```
sage: (2*atan(1/2) - atan(1/7) - pi/4).full_simplify()
-1/4*pi + 2*arctan(1/2) - arctan(1/7)
```

Дело в том, нет алгоритма, позволяющего выяснить, равно ли заданное символьное выражение нулю или нет.

Еще одна символьная операция — подстановка, которая позволяет заменить в выражении переменную на некоторое другое выражение.

```
sage: sin(x^2+pi).subs(x=1)
sin(pi + 1)
sage: sin(x^2+pi).subs(x==1)
sin(pi + 1)
sage: sin(x^2*y+pi).subs([x==1,y==3])
sin(pi + 3)
sage: sin(x^2+pi).subs(x==y+2)
sin(pi + y^2 + 4*y + 4)
```

В качестве аргумента `subs` имеет или равенство, или список равенств. Мы рекомендуем использовать `==`, чтобы рассматривать подстановку как равенство.

5. Задания

Теоретические задания.

1) Дайте определение кольца.
2) Дайте определение поля.
3) Элемент e кольца A называется единицей, если $ae = ea = a$. Докажите, что $e = 1$.
4) Является ли реализация поля вещественных чисел в Sage кольцом? Полем?

Практические задания.

1) Преобразуйте π^2 в десятичную дробь.
2) Совпадает ли `QQ(RR(pi))` с числом π? Совпадает ли `QQ(RR(1/3))` с числом $\frac{1}{3}$?

3) Как интерпретирует Sage числа, введенные с клавиатуры как `1/3` и `1.0/3`? Совпадают ли `1/3` и `1.0/3` как элементы поля $\mathbb{Q}$?

4) Упростите выражение $5(n-2)^3 - 6(n+3)^3 - 3(2n-9)^3$.

5) Подставьте $x = \pi$ в выражение $\sin(x^2 + 2x) + x^3$ и вычислите его приближенно.

6) Подставьте $x = y^2+3$ в выражение x^2+y^2-xy и упростите полученный результат.