УДК 512.558

# Компьютерное нахождение четырехэлементных мультипликативно идемпотентных полуколец

**Р. А. Михеев, А. А. Петров**
Вятский государственный университет; Киров, Россия
**e-mail:** miheev.roma90@mail.ru;
**e-mail:** apetrov43@mail.ru; **ORCID: 0000-0002-5877-2850, AuthorID: 662310**

Описывается компьютерная программа для поиска всех четырехэлементных мультипликативно идемпотентных полуколец. Установлено, что с точностью до изоморфизма таких полуколец ровно 381, они представлены таблицами Кэли аддитивных и мультипликативных редуктов. Приведены необходимые определения, свойства и иллюстрации.

**Ключевые слова:** *конечное полукольцо; идемпотентность; мультипликативно идемпотентное полукольцо; компьютерное моделирование.*

*Поступила в редакцию 30.03.2022, принята к опубликованию 25.04.2022*

# Computer Finding of Four-element Multiplicatively Idempotent Semirings

**R. A. Mikheev, A. A. Petrov**
Vyatka State University; Kirov, Russia
**e-mail:** miheev.roma90@mail.ru;
**e-mail:** apetrov43@mail.ru; **ORCID: 0000-0002-5877-2850; AuthorID: 662310**

In this paper we describe a computer program for searching for all four-element multiplicatively idempotent semirings. We have established that up to the isomorphism of such semirings exactly 381, they are represented by Cayley tables of additive and multiplicative reducts. We give the necessary definitions, properties and illustrations.

**Keywords:** *finite semiringi; dempotence; multiplicatively idempotent semiring; computer modeling.*

*Received 30.03.2022, accepted 25.04.2022*

DOI: 10.17072/1993-0550-2022-2-46-52

Данная статья является продолжением работы [2], в которой вручную найдены все трехэлементные полукольца с идемпотентным умножением и описаны их свойства.

В работе [6] с помощью компьютера описаны все трехэлементные аддитивно идемпотентные полукольца, с точностью до изоморфизма (таких полуколец ровно 61).

Общая теория полуколец изложена в монографии Голана [4]. Мультипликативно идемпотентным полукольцам посвящены работы [1–3, 5].

*Полукольцом* называется алгебраическая структура $\langle S, +, \cdot \rangle$ с коммутативно-ассоциативной операцией сложения + и ассоциативной операцией умножения, дистрибутивной относительно сложения с обеих сторон.

Полукольцо $S$ называется:

– *коммутативным*, если $S$ удовлетворяет тождеству $xy=yx$;

– *мультипликативно идемпотентным* (*аддитивно идемпотентным*), если на нем тождественно $xx=x$ (соответственно, $x+x=x$);

– *идемпотентным*, если $S$ одновременно мультипликативно и аддитивно идемпотентное;

– *моно-полукольцом*, если $x+y=xy$ для всех $x, y \in S$;

– *полукольцом с константным сложением*, если оно удовлетворяет тождеству $x+y=u+v$.

Элемент $\theta$ произвольного полукольца $S$ назовем *поглощающим по умножению* (*поглощающим по сложению*), если для всех $x \in S$ выполняется $\theta \cdot x = x \cdot \theta = \theta$ (соответственно, $x+\theta=\theta$). Элемент $\infty \in S$, поглощающий по сложению и по умножению, называется *поглощающим*.

Если в полукольце $S$ существует элемент 0, нейтральный по сложению и поглощающий по умножению, то $S$ называется *полукольцом с нулем* 0. Если же полукольцо $S$ обладает элементом 1, нейтральным по умножению, то $S$ называется *полукольцом с единицей* 1.

Отметим, что к любому полукольцу $S$ можно естественным образом присоединить нулевой элемент 0 или поглощающий элемент $\infty$. Обозначим полученные полукольца $S \cup \{0\}$ и $S \cup \{\infty\}$, соответственно.

Хорошо известно, что с точностью до изоморфизма существует ровно шесть двухэлементных мультипликативно идемпотентных полуколец:

- двухэлементная цепь $\mathbf{B}=\{0,1\}$;
- двухэлементное поле $\mathbf{Z}_2=\{0,1\}$;
- двухэлементное идемпотентное моно-полукольцо $\mathbf{D}=\{1,\infty\}$ с единицей 1;
- двухэлементное полукольцо $\mathbf{T}=\{1,\infty\}$ с единицей 1 и константным сложением ($x+y=\infty$);
- двухэлементное идемпотентное полукольцо $\mathbf{L}=\{a,b\}$, с тождеством $xy=x$;
- двухэлементное идемпотентное полукольцо $\mathbf{R}=\{a,b\}$, с тождеством $xy=y$.

Для любого полукольца $\langle S, +, \cdot \rangle$ существует антиизоморфное полукольцо $\langle S, +, * \rangle$, в котором тождественно $x * y = y \cdot x$; такое *дуальное* полукольцо обозначим через $S^*$.

Полукольцо $S$, для которого $S^* \cong S$, назовем *самодуальным*. Ясно, что любое коммутативное полукольцо $S$ самодуально. В работе [2] установлено, что среди трехэлементных некоммутативных мультипликативно идемпотентных полуколец самодуальных нет.

**Пример 1.** Положим $S=\mathbf{L}\times\mathbf{R}$. Умножение на некоммутативном четырехэлементном мультипликативно идемпотентном полукольце $S$ выполняется следующим образом:

$$(x, y) \cdot (u, v) = (xu, yv) = (x, v)$$

для любых $x, y, u, v \in \{a, b\}$.

Соответственно, сложение на полукольце $S^*$, дуальном к $S$, задается так же, как на $S$, а умножение определяется правилом:

$$(x, y) * (u, v) = (u, v) \cdot (x, y) = (u, y),$$

и, как легко видеть, $S^* \cong \mathbf{R}\times\mathbf{L} \cong \mathbf{L}\times\mathbf{R} = S$, то есть $S$ – некоммутативное самодуальное мультипликативно идемпотентное полукольцо.

Для описания всех четырехэлементных мультипликативно идемпотентных полуколец $S$ нами составлена компьютерная программа на языке программирования С.Опишем схему создания этой программы.

Аддитивный и мультипликативный редукты полукольца задаются с помощью таблиц Кэли. Таблицы представляют собой одномерные массивы matrix длины $N^2$ из элементов 0, …, N–1, в которых с 0-ого по (N–1)-ую позицию занимает первая строка таблицы, с N-ного по (2N–1)-й – вторая строка таблицы и т. д. Соответственно, позиция элемента в строке i и столбце j в массиве matrix равна i*N + j.

### Шаг 1. Генерация ассоциативных таблиц

*//проверка таблицы на ассоциативность*

```
bool isassociative(int matrix[N*N]) {
   for (int i = 0; i < N; i++) {
  for (int j = 0; j < N; j++) {
   for (int k = 0; k < N; k++) {
    if (matrix[matrix[i*N + j]*N + k] != matrix[i*N + matrix[j*N + k]])
     return false;
   }
  }
 }
 return true;
}
```

```
//Создание аддитивной полугруппы полукольца S с учетом ее коммутативности
void generate_commutative_tables(List* mult_tables) {
 int matrix[N*N];
 generate_commutative_tables_rec(matrix, 0, 0, mult_tables);
}
void generate_commutative_tables_rec(int matrix[N*N], int row_pos, int col_pos, List* add_tables) {
 if (row_pos == N && col_pos == 0) {
  if (isassociative(matrix)) {
   int* matrix_copy = copy_matrix(matrix);
   list_push(add_tables, matrix_copy);
  }
  return;
 }
 for (int i = 0; i < N; i++) {
  matrix[row_pos*N + col_pos] = i;
  matrix[col_pos*N + row_pos] = i;
  int new_row_pos = row_pos;
  if (col_pos == N - 1)
   new_row_pos++;
  int new_col_pos = (col_pos + 1 + new_row_pos) % N;
  generate_commutative_tables_rec(matrix, new_row_pos, new_col_pos, add_tables);
 }
}

//Создание мультипликативно идемпотентной полугруппы
void generate_idempotent_tables(List* mult_tables) {
 int matrix[N*N];
 for (int j = 0; j < N; j++) {
  matrix[j*N + j] = j;
 }
 generate_idempotent_tables_rec(matrix, 0, 1, mult_tables);
}
void generate_idempotent_tables_rec(int matrix[N*N], int row_pos, int col_pos, List* mult_tables) {
 if (row_pos == N && col_pos == 0) {
  if (isassociative(matrix)) {
   int* matrix_copy = copy_matrix(matrix);
   list_push(mult_tables, matrix_copy);
  }
  return;
 }
 for (int i = 0; i < N; i++) {
  matrix[row_pos*N + col_pos] = i;
  int new_col_pos = (col_pos + 1) % N;
  int new_row_pos = row_pos;
  if (new_col_pos == 0)
   new_row_pos++;
  if (new_col_pos == new_row_pos) {
   new_col_pos = (new_col_pos + 1) % N;
   if (new_col_pos == 0)
    new_row_pos++;
  }
  generate_idempotent_tables_rec(matrix, new_row_pos, new_col_pos, mult_tables);
 }
}
```

### Шаг 2. Проверка дистрибутивности умножения относительно сложения для построенных таблиц Кэли

```
//Проверка левой дистрибутивности
bool isdistributive(int mult[N*N], int add[N*N]) {
 return isdistributive_left(mult, add) && isdistributive_right(mult, add);
}
bool isdistributive_left(int mult[N*N], int add[N*N]) {
 for (int i = 0; i < N; i++) {
  for (int j = 0; j < N; j++) {
   for (int k = 0; k < N; k++) {
    if (mult[i*N + add[j*N + k]] != add[mult[i*N + j]*N + mult[i*N + k]])
     return false;
   }
  }
 }
 return true;
}
```

bool isdistributive_right – правая дистрибутивность проверяется аналогично левой

```
//Проверка коммутативности умножения. Если умножение коммутативно, проверяем только левую дистрибутивность умножения относительно сложения.
bool iscommutative(int table[N*N]) {
 for (int i = 0; i < N; i++) {
  for (int j = i; j < N; j++) {
   if (table[i*N + j] != table[j*N + i])
    return false;
  }
 }
 return true;
}
```

```
void generate_semirings(List* semirings, List mult_tables, List add_tables) {
 struct Node* mult_temp = mult_tables.head;
 while (mult_temp != NULL) {
  bool iscomm = iscommutative(mult_temp->value);
  struct Node* add_temp = add_tables.head;
  while (add_temp != NULL) {

   if (iscomm && isdistributive_left(mult_temp->value, add_temp->value)
    || !iscomm && isdistributive(mult_temp->value, add_temp->value)) {
    Semiring* semiring = (Semiring*)malloc(sizeof(Semiring));
    semiring->mult = mult_temp->value;
    semiring->add = add_temp->value;
    list_push(semirings, semiring);
   }
   add_temp = add_temp->next;
  }
  mult_temp = mult_temp->next;
 }
}
```

### Шаг 3. Выбор из полученного списка неизоморфных полуколец

```
//Генерируем одномерные массивы – перестановки чисел 0, …, N–1.
void generate_arrays(List* arrays) {
 int array[N];
 generate_arrays_rec(array, 0, arrays);
}
void generate_arrays_rec(int arr[N], int pos, List* arrays) {
 if (pos == N) {
  list_push(arrays, copy_array(arr));
 } else {
  for (int i = 0; i < N; i++) {
   bool found = false;
   for (int j = 0; j < pos; j++) {
    if (arr[j] == i) {
     found = true;
     break;
    }
   }
   if (found)
    continue;
   arr[pos] = i;
   generate_arrays_rec(arr, pos + 1, arrays);
  }
 }
}
//Перебираем пары полуколец, проверяя свойства f(a·b) = f(a)·f(b), f(a+b) = f(a)+f(b) для произвольной подстановки f на множестве {0, 1, …, N–1}. Если для некоторой пары полуколец s1 и s2 существует f с перечисленными свойствами, то s1 и s2 изоморфны.
bool isisomorphism(int f[N], Semiring s1, Semiring s2) {
 for (int a = 0; a < N; a++) {
  for (int b = 0; b < N; b++) {
   if (f[s1.mult[a*N + b]] != s2.mult[f[a]*N + f[b]]
    || f[s1.add[a*N + b]] != s2.add[f[a]*N + f[b]])
    return false;
  }
 }
 return true;
}
bool areisomorphic(Semiring s1, Semiring s2, List arrays) {
 bool result = false;
 struct Node* temp = arrays.head;
 while (temp != NULL) {
  if (isisomorphism(temp->value, s1, s2))
   return true;
  temp = temp->next;
 }
 return false;
}
//Перебираем список полученных полуколец и исключаем изоморфные полукольца
void filter_isomorphism(List* semirings, List arrays) {
 struct Node* temp = semirings->head;
 while (temp != NULL) {
  struct Node* temp_inner = temp->next;
  while (temp_inner != NULL) {
   if (areisomorphic(*((Semiring*)temp->value), *((Semiring*)temp_inner->value), arrays)) {
    struct Node* temp_next = temp_inner->next;
    list_delete(semirings, temp_inner);
    temp_inner = temp_next;
   } else temp_inner = temp_inner->next;
  }
  temp = temp->next;
 }
}
```

### Шаг 4. Проверка свойств найденных мультипликативно идемпотентных полуколец

```
//идемпотентность
bool isidempotent(int matrix[N*N]) {
 for (int i = 0; i < N; i++) {
  if (matrix[i*N + i] != i)
   return false;
 }
 return true;
}
//наличие нейтрального элемента 1 относительно умножения
int neutral(int matrix[N*N]) {
 for (int i = 0; i < N; i++) {
  if (isneutral(matrix, i)) {
   return i;
  }
 }
 return -1;
}
bool isneutral(int matrix[N*N], int inx) {
 for (int i = 0; i < N; i++) {
  if (matrix[inx*N + i] != i || matrix[i*N + inx] != i) {
   return false;
  }
 }
 return true;
}
//наличие нейтрального элемента 0 относительно сложения
int zero(int mult[N*N], int add[N*N]) {
 int add_neu = neutral(add);
 if (add_neu == -1) {
  return -1;
 }
 for (int i = 0; i < N; i++) {
  if (mult[add_neu*N + i] != add_neu || mult[i*N + add_neu] != add_neu) {
   return -1;
  }
 }
 return add_neu;
}
//наличие поглощающего элемента
int infinity(int mult[N*N], int add[N*N]) {
 for (int i = 0; i < N; i++) {
  if (isinfinity(mult, add, i)) {
   return i;
  }
 }
 return -1;
}
bool isinfinity(int mult[N*N], int add[N*N], int inx) {
 for (int i = 0; i < N; i++) {
  if (
   mult[i*N + inx] != inx || mult[inx*N + i] != inx
   ||
   add[i*N + inx] != inx || add[inx*N + i] != inx
  ) {
   return false;
  }
 }
 return true;
}
//проверка константности сложения
bool isconst(int table[N*N]) {
   for (int i = 0; i < N*N - 1; i++) {
      if (table[i] != table[i + 1]) {
         return false;
      }
   }
   return true;
}
//проверка, является ли полукольцо моно-полукольцом
bool ismono(int mult[N*N], int add[N*N])
{
 for (int i = 0; i < N*N; i++) {
  if (mult[i] != add[i]) {
   return false;
  }
 }
 return true;
}
```

### Шаг 5. Переобозначение элементов полукольца

Обозначаем элементы полученных полуколец через *a*, *b*, *c*, *d* и т. д. Нулевой, единичный и поглощающий элементы обозначаем как 0, 1 и *i*, соответственно. Полученный результат записывается в текстовый файл (рис. 1):

```
                comm    idem    mono    const   zero    one     inf
add     mult    +       -       -       +       -       -       +
i i i i  i i i i
i i i i  i a i i
i i i i  i i b i
i i i i  i i i c

add     mult    +       -       -       -       -       -       +
i i i i  i i i i
i i i i  i a i i
i i i i  i i b i
i i i c  i i i c

add     mult    +       -       -       -       -       -       -
a a c a  a a a a
a a c a  a b a a
c c a c  a a c a
a a c a  a a a d
```

**Рис. 1**

Полный исходный код описанной программы доступен по ссылке:

https://clck.ru/gMyDR

При N=3 получен результат из [2, теорема 1]: с точностью до изоморфизма существует ровно 43 трехэлементных мультипликативно идемпотентных полукольца, среди которых:

- 19 коммутативных полуколец, из них 7 идемпотентных;
- 23 идемпотентных полукольца, в том числе 16 некоммутативных;
- 14 полуколец с единицей 1, 6 полуколец с нулем 0, 4 полукольца имеют одновременно 0 и 1;
- 12 полуколец с поглощающим элементом ∞, 5 полуколец с 1 и ∞.

При N=4 результатом работы программы стала следующая

**Теорема 1.** *С точностью до изоморфизма существует ровно 381 четырехэлементное мультипликативно идемпотентное полукольцо. Среди этих полуколец:*

- *118 коммутативных, из них 33 идемпотентны;*
- *166 идемпотентных;*
- *46 полуколец с нулем 0;*
- *67 полуколец с единицей 1;*
- *17 полуколец с 0 и 1;*
- *78 полуколец с поглощающим элементом ∞, из них 20 имеют 1;*
- *15 полуколец с константным сложением;*
- *5 полуколец являются монополукольцами.*

**Замечание 1.** Время работы созданной программы для поиска всех четырехэлементных мультипликативно идемпотентных полуколец на персональном компьютере с процессором AMD FX-4300 составляет 2,73 секунды. Следует заметить, что программа основана на большом переборе вариантов, поэтому уже при N=5 она не дает требуемый результат за разумное время.

В дальнейшем планируется усовершенствование алгоритма для сокращения времени ее работы.

**Замечание 2.** Мультипликативная полугруппа полукольца с коммутативным идемпотентным умножением является *нижней полурешеткой*, то есть частично упорядоченным множеством ($a\le b\Leftrightarrow ab=a$), для каждой пары $a, b$ элементов которого имеется точная нижняя грань $\inf\{a,b\}=ab$. С точностью до изоморфизма существует 5 четырехэлементных нижних полурешеток (рис. 2):

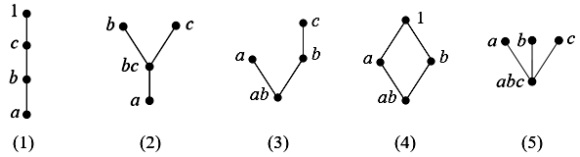


**Рис. 2**

Из 118 полуколец с коммутативным идемпотентным умножением 37 полуколец с умножением (1), 27 полуколец с умножением (2), 34 полукольца с мультипликативной полурешеткой (3) и по 10 полуколец типа (4) и (5).

Заметим, что с точностью до изоморфизма существует ровно 15 пятиэлементных нижних полурешеток. Поэтому поиск всех неизоморфных пятиэлементных коммутативных мультипликативно идемпотентных полуколец можно начинать с подбора аддитивной коммутативной полугруппы для каждой из нижних полурешеток по умножению.

**Замечание 3.** Присоединяя внешним образом к полученным в теореме 1 четырехэлементным полукольцам нулевой элемент 0 (поглощающий элемент ∞), получим 381 пятиэлементное мультипликативно идемпотентное полукольцо с 0 (соответственно, с ∞).

Пусть $S$ – произвольное пятиэлементное мультипликативно идемпотентное полукольцо с нулем 0. Известно [1, теорема 4.1.1], что $S$ представимо в виде прямой суммы некоторых булева кольца $R$ и антикольца $T$ (напомним, что *антикольцом* называется полукольцо, удовлетворяющее квазитождеству $x+y=0\Rightarrow x=0$). Но любое неодноэлементное конечное булево кольцо изоморфно прямому произведению двухэлементных полей, а потому имеет четное число элементов. Значит, $R=\{0\}$, и $S=T$ – антикольцо. Если при этом в $S$ нет делителей нуля (то есть $ab\neq 0$ для любых $a, b\in S\setminus\{0\}$), то $S\setminus\{0\}$ будет четырехэлементным полукольцом.

Стало быть, нахождение всех пятиэлементных мультипликативно идемпотентных полуколец $S$ с нулем сводится к случаю, когда $S$ имеет ненулевые делители нуля.

*Авторы выражают благодарность профессору Е. М. Вечтомову за постановку задачи и внимание к работе.*

## Список литературы


1. *Вечтомов Е.М., Петров А.А.* Полукольца с идемпотентным умножением. Киров: Изд-во ООО "Радуга-ПРЕСС", 2015. 144 с.
2. *Вечтомов Е.М., Петров А.А.* Трехэлементные мультипликативно идемпотентные полукольца // Математический вестник Вятского государственного университета. 2021. № 2 (21). С 13–23.
3. *Chaida I., Länger H., Švrček F.* Multiplicatively idempotent semirings // Mathematica Bohemica. 2015. Vol. 140, № 1. P. 35–42.
4. *Golan J. S.* Semirings and their applications. Kluwer Academic Publishers: Dordrecht-Boston-London, 1999. 380 p.
5. *Vechtomov E.M., Petrov A.A.* Multiplicatively Idempotent Semirings // Journal of Mathematical Sciences (New York). 2015. Vol. 206. Issue 6. P. 634–653.
6. *Zhao X., Ren M., Crvenković S., Shao Y., Dapić P.* The variety generated by an ai-semiring of order three // Ural Mathematical Journal. 2020. Vol. 6. Issue 2. P. 117–132.


## References


1. *Vechtomov E. M., Petrov A. A.* Polukol'ca s idempotentnym umnozheniem. Kirov: Izd-vo OOO "Raduga-PRESS", 2015. 144 s.
2. *Vechtomov E. M., Petrov A. A.* Trekhelementnye mul'tiplikativno idempotentnye polukol'ca // Matematicheskij vestnik Vyatskogo gosudarstvennogo universiteta. 2021. № 2 (21). S. 13–23.
3. *Chaida I., Länger H., Švrček F.* Multiplicatively idempotent semirings // Mathematica Bohemica. 2015. Vol. 140, № 1. P. 35–42.
4. *Golan J. S.* Semirings and their applications. Kluwer Academic Publishers: Dordrecht-Boston-London, 1999. 380 p.
5. *Vechtomov E.M., Petrov A.A.* Multiplicatively Idempotent Semirings // Journal of Mathematical Sciences (New York). 2015. Vol. 206. Issue 6. P. 634–653.
6. *Zhao X., Ren M., Crvenković S., Shao Y., Dapić P.* The variety generated by an ai-semiring of order three // Ural Mathematical Journal. 2020. Vol. 6. Issue 2. P. 117–132.



**Просьба ссылаться на эту статью:**

*Михеев Р.А., Петров А.А.* Компьютерное нахождение четырехэлементных мультипликативно идемпотентных полуколец // Вестник Пермского университета. Математика. Механика. Информатика. 2022. Вып. 2(57). С. 46–52. DOI: 10.17072/1993-0550-2022-2-46-52.

**Please cite this article as:**

*Miheev R.A., Petrov A.A.* Computer Finding of Four-element Multiplicatively Idempotent Semirings // Bulletin of Perm University. Mathematics. Mechanics. Computer Science. 2022. Issue 2(57). P. 46–52. DOI: 10.17072/1993-0550-2022-2-46-52.