Дискретная математика. Коллоквиум весна 2017. Задачи

Ваномас

11 марта 2017 г.

Задача 1

Вероятностное пространство: последовательности (x1, x2, x3, x4) длины 4, состоящие из целых чисел в диапазоне от 1 до 6. Все исходы равновозможны. Найдите вероятность события x1x2x3x4 четно.

Решение

Если хотя бы одно из чисел четно, то всё произведение тоже четно. Четных исходов от 1 до 6: 2, 4, 6. Всего исходов: 6^4. интересующих нас исходов(выбираем от одной до 4 позиций, где будут стоять четные числа, на остальные ставим нечетные) $4*3*3^3+6*3*3*3^2+4*3*3^3+3^4 = 1215$.

Ответ: $1215/1296 = 0.9375$.

Задача 3

Существуют ли такие события A и B, что $P[A] = P[B] = P[A|B] = \frac{1}{2}$ Pr[B|A] $= \frac{1}{3}$

Решение

По формуле для условной вероятности, $Pr[A|B] = \frac{Pr[AB]}{Pr[B]}$, где Pr[AB] - вероятность наступления двух событий сразу. Также $Pr[B|A] = \frac{Pr[BA]}{Pr[A]}$, По определению, $Pr[BA] = P[AB]$, а по условию $Pr[B] = P[A]$. Из этого следует, что $Pr[B|A] = Pr[A|B]$, что противоречит условию

Ответ: не существуют.

Задача 4

О событиях A и B вероятностного пространства U известно, что Pr$[A]$ = Pr$[B]$ = 4/5. Могут ли при этом события $A \cup B$ и B быть независимыми?

Решение

Пусть вероятностное пространство U задано таким образом, что любой исход на нём — число от 0 до 4, событие A: «число не равно 0» и B: «число не равно 1». Очевидно, что вероятность каждого из этих событий в отдельности равна $4/5$. Событию $A \cup B$: «число не равно 0 *или* число не равно 1» удовлетворяют все исходы. Таким образом, выполнено равенство $\Pr[A \cup B] \cdot \Pr[B] = 1 \cdot \Pr[B] = \Pr[B]$, то есть события $A \cup B$ и B являются независимыми.

Задача 7

Докажите, что всякое бесконечное множество содержит бесконечное число непересекающихся счетных подмножеств.

Решение

Известно, что для любого бесконечного множества существует счетное подмножество, тогда уберем из данного бесконечного множества A счетное подмножество B. Надо выбрать счётное множество, и его разбить на счётное количество счётных. А это делается просто на основании того факта, что множество клеток координатного угла счётно (это хорошо известный факт). Поэтому нумеруем все упорядоченные пары вида (i,j), где i,jeN, и сопоставляем их элементам счётного подмножества B. Потом разбиваем его на счётное число счётных подмножеств вида Bj, где элементам Bj соответствуют пары вида (i,j) с фиксированным j и произвольными $i >=$ 1.

Задача 18

Постройте вычислимую биекцию между множествами $\mathbb{N}$ и $\mathbb{N} \setminus \{p^2 \mid p \in \mathbb{N}\}$.

Решение

Сначала определим разрешающую функцию второго множества $f(n)$:

```
i = 0
while (i * i ) < n
  i = i + 1
return (i * i == n)
```

Теперь рассмотрим отображение $g(x) : \mathbb{N} \to \mathbb{N} \setminus \{p^2 \mid p \in \mathbb{N}\}$, заданное следующим образом:

```
i = 0
j = 0
while (i != n)
```

```
    while (f(j))
      j = j + 1
    i = i + 1
  return j
```

Данное отображение сопоставляет числу x элемент множества $\mathbb{N} \setminus \{p^2 \mid p \in \mathbb{N}\}$, имеющий номер x с начала в порядке возрастания (элементы $\mathbb{N} \setminus \{p^2 \mid p \in \mathbb{N}\}$ могут быть перечислены в порядке возрастания, так как оно разрешимо). Оно инъективно, так как элементы с различными номерами не равны (два различных элементы множества не равны). Также оно сюръективно, поскольку любой элемент $\mathbb{N} \setminus \{p^2 \mid p \in \mathbb{N}\}$ имеет некоторый номер, и для элемента из этого множества с номером n прообразом будет число n. Таким образом, отображение g биективно.

Задача 19

Постройте вычислимую биекцию между множеством двоичных слов и натуральными числами.

Решение

Выпишем в столбец все двоичные слова длины 1, затем в ещё один столбец — двоичные слова длины 2, затем — слова длины 3, и так далее. Пронумеруем последовательно, начиная с 0, сначала слова в первом столбце, затем, продолжая нумерацию, во втором, затем — в третьем, и так далее. Нумерация будет выглядеть следующим образом:

0 : **0**	2 : **00**	6 : **000**	14 : **0000**	.....
1 : **1**	3 : **01**	7 : **001**	15 : **0001**	.....
	4 : **10**	8 : **010**	16 : **0010**	.....
	5 : **11**	9 : **011**	17 : **0011**	.....
		10 : **100**	18 : **0100**	.....
		11 : **101**	19 : **0101**	.....
		12 : **110**	20 : **0110**	.....
		13 : **111**	21 : **0111**	.....
			22 : **1000**	.....
			23 : **1001**	.....
			24 : **1010**	.....
			25 : **1011**	.....
			26 : **1100**	.....
			27 : **1101**	.....
			28 : **1110**	.....
			29 : **1111**	.....

Посроим отображение, которое ставит в соответствие натуральному числу n двоичное слово, имеющее номер n в данной нумерации. Заметим, что это отображение инъективно — слова с различными номерами различны, так как слова в одном столбце различны по построению, а слова в различных столбцах имеют разную длину. Также оно сюръективно, поскольку каждому номеру соответствует некоторое двоичное слово, так как их бесконечно много, и они были

пронумерованы последовательно (если какому-то номеру не соответствует двоичное слово, значит, либо пронумеровано конечное количество слов, либо какой-то номер был «пропущен»). Таким образом, построенное отображение биективно.

Задача 20

Пусть f — вычислимая биекция между $\mathbb{N}$ и $\mathbb{N}$. Докажите, что обратная биекция f^{-1} также вычислима.

Решение

Функция $f(x) : \mathbb{N} \to \mathbb{N}$ — вычислимое биективное отображение. Рассмотрим следующий алгоритм, принимающий на вход x:

```
i = 0
while (f(i) != x)
  i = i + 1
return i
```

Этот алгоритм, очевидно, на входе x вернёт значение i, на котором f принимает значение x, то есть $f^{-1}(x)$. Также цикл `while` совершит ровно i итераций, то есть конечное число, поскольку прообраз x конечен. Таким образом, алгоритм остановится на любом входе за конечное число шагов, то есть соответствующая ему функция f^{-1} вычислима.

Задача 21

Докажите, что, если функция f вычислима и $A \subset \mathbb{N}$ — перечислимое множество, то и образ, и прообраз множетсва A перечислимы.

Решение

Докажем, что декартово произведение перечислимых множеств также перечислимо. Пусть множества A и B перечислимы, то есть для них существуют алгоритмы, с помощью которых могут быть получены все их элементы. Из этого следует, что элементы этих множеств могут быть пронумерованы в порядке, в котором они выводятся вышеназванными алгоритмами. Тогда для них существуют функции $f(x)$ и $b(x)$, сопоставляющие натуральному числу x элементы A и B соответственно, имеющие в вышеобозначенной нумерации номер x (если множество конечно, то для x, больших числа элементов в этом множестве, соответствующая функция не определена). Предположим, что оба множества бесконечны. Тогда рассмотрим следующий алгоритм, принимающий на вход натуральное число x:

```
a_index = 0
b_index = 0
number = 0
while (number != x)
  if (a_index > 0)
    a_index = a_index - 1
    b_index = b_index + 1
  else
    a_index = b_index + 1
    b_index = 0
  numder = number + 1
return (a[a_index], b[b_index])
```

Докажем по индукции по $(i + j)$, что комбинация элементов $(a[i], b[j])$ будет выведена на входе $\frac{(i+j)(i+j+1)}{2} + j$. Заметим, что на входе x цикл `while` совершает ровно x итераций. База индукции — $(i+j) = 0$. Такому условию удовлетворяет только комбинация $(a[0], b[0])$, и несложно проверить, что она действительно будет выведена на входе 0. Пусть теперь утверждение верно для всех комбинаций, для которых $(i + j) = k$. Докажем это для комбинаций, для которых $(i + j) = k + 1$. По предположению индукции на входе $\frac{k(k+1)}{2} + k$ будет выведена комбинация $(a[0], b[k])$, то есть через $\frac{k(k+1)}{2} + k$ итерацию значениея `a_index` и `b_index` будут соответственно равны 0 и k. Тогда на следующей итерации значения станут равны соответственно $k + 1$ и 0. Как мы знаем, цикл остановится на следующей итерации на входе $\frac{k(k+1)}{2} + k + 1$, то есть $\frac{(k+1)(k+2)}{2} + 0$. На следующем $k + 1$ входе финальное значение `a_index` будет каждый раз на 1 уменьшатья, а `b_index` — увеличиваться, то есть на входе $\frac{(k+1)(k+2)}{2} + j$ будет выведена комбинация $(a[k + 1 - j], b[j])$ (при $j \geqslant k + 1$, что и требовалось доказать.

Теперь рассмотрим случай, в котором хотя бы одно из множеств конечно. Без ограничения общности предположим, что это множество A. Пусть количество его элементов равно A_size. Тогда алгоритм, получающий все его элементы, совершает при этом конечное число шагов (так как для получения одного элемента он должен совершать конечное число шагов — иначе этот элемент никогда не будет получен). В таком случае рассмотрим следующий алгоритм, принимающий на вход натуральное число x:

```
a_index = 0
b_index = 0
number = 0
while (number != x)
  if (a_index < A_size)
    a_index = a_index + 1
  else
    a_index = 0
    b_index = b_index + 1
  number = number + 1
return (a[a_index], b[b_index])
```

Этот алгоритм выведет комбинацию $(a[i], b[j])$ на входе $j \cdot A\_size + i$ (в предположении, что $i \leqslant A\_size$) Это несложно доказать: заметим, что каждые A_size итераций цикла `while` значение `a_index` обнуляется, а `b_index` увеличивается на 1, в остальных же итерациях `a_index`

увеличивается на 1, то есть финальное значение `a_index` равно остатку при делении количества выполненных итераций на A_size, а значение `b_index` — целой части при делении на A_size. Количество же итераций, по аналогии с предыдущим описанным алгоритмом, равно входному значению. Заметим также, что бесконечность множества B нигде не использовалась: если оно конечно, то вышеописанный алгоритм никогда не остановится на входах $j \cdot A\_size + i$, где $j > B\_size$, поскольку во время его работы будет вызвана невычислимая функция $b[j]$.

Таким образом, выше было показано, что декартово произведение двух перечислимых множеств перечислимо. Докажем теперь, что, если функция f вычислима и $A \subset \mathbb{N}$ — перечислимое множество, то прообраз A перечислим. Поскольку A перечислимо, то и $A \times \mathbb{N}$ также перечислимо. Тогда существует алгоритм $\mathfrak{A}$, получающий все элементы этого множества. Рассмотрим алгоритм, который запускает $\mathfrak{A}$, и, получая некоторый элемент (x, y), проверяет, является ли x образом y при отображении f, и, если это так, выводит y. Поскольку x — элемент A, то, что x является образом y, означает, что y входит в прообраз A, то есть вышеописанный алгоритм выводит только элементы $f^{-1}(A)$. Докажем, что он выведет все элементы $f^{-1}(A)$: если m входит в $f^{-1}(A)$, существует n такой, что $n \in A$ и $f(m) = n$. Поскольку $\mathfrak{A}$ получает все пары (x, y), где $x \in A$ и $y \in \mathbb{N}$, за конечное число шагов, а функция f вычислима (то есть алгоритм завершит «обработку» любой полученной $\mathfrak{A}$ пары за конечное число шагов), пара n, m также будет получена, а, поскольку $f(m) = n$, число m будет выведено.

Теперь докажем, что, если функция f вычислима и $A \subset \mathbb{N}$ — перечислимое множество, то образ A также перечислимое множество. Поскольку A перечислимо, существует некий алгоритм $\mathfrak{T}$, получающий все элементы A. Рассмотрим алгоритм, который запускает $\mathfrak{T}$ и, получая элемент x, выводит $f(x)$. Этот алгоритм, очевидно, выводит только элементы из $f(A)$. Также любой элемент $f(A)$ будет выведен через конечное число шагов, так как для любого $a \in f(A)$ существует $x \in A$ такое, что $a = f(x)$, $\mathfrak{T}$ получает любой элемент A за конечное число шагов, а f вычислима.

Задача 22

Найдите разрешимое множество A и вычислимую функцию f такие, что прообраз $f^{-1}(A)$ неразрешим.

Решение

Пусть K — некоторое перечислимое, но неразрешимое множество, $k(x)$ — функция, возвращающая x-ый по счёту вывод перечисляющего алгоритма $\mathfrak{A}$ для множества K. Тогда рассмотрим следующий алгоритм, принимающий на вход натуральное число x:

```
i = 0
while (true)
  if (k(i) == x)
    return 1
  i = i + 1
```

Данный алгоритм, по сути, запускает внутри себя $\mathfrak{A}$, последовательно для каждого вывода $\mathfrak{A}$ проверяет, не равен ли данный вывод x, и в случае положительного ответа останавливается, выводя «1» как результат своей работы. Поскольку для любого $k \in K$ алгоритм $\mathfrak{A}$ выводит k через конечное число шагов, на входе k вышеописанный алгоритм завершиться через конечное число шагов. На любом входе $p \notin K$ алгоритм никогда не завершится, поскольку p не является результатом работы $\mathfrak{A}$, и, следовательно, не принадлежит множеству значений $k(x)$. Функция f, вычисляемая приведённым выше алгоритмом, возвращает 1 на элементах K и не определена на $\mathbb{N} \setminus K$. Таким образом, $f^{-1}(\{1\}) = K$, то есть f и $\{1\}$ являются соответственно искомой функцией и искомым множеством.

Задача 29

Приедите пример машины Тьюринга, вычисляющей нигде не определённую функцию.

Решение

Рассмотрим машину Тьюринга, работающую на ленте с унарным алфавитом $\{a\}$ (любые входные данные могут быть представлены в унарной системе) и заданную следующим набором правил:

	$\boldsymbol{a}$	$\boldsymbol{\Lambda}$
$\boldsymbol{q_0}$	q_0, a, -1	q_0, Λ, 1

Для неё возможны только две комбинации состояния и значения под считывающей головкой: (q_0, a) и (q_0, Λ). Для обоих этих состояний заданы правила, следовательно, машина никогда не остановится, и вычисляемая ею функция нигде не будет определена.

Задача 30

Постройте машину Тьюринга, находящую неполное частное от деления на 3 в унарной системе.

Решение

Пусть машина Тьюринга в начале своей работы находится в конфигурации $\langle \Lambda, q_0, x \rangle$ (где x — входные данные в унарной системе) и задаётся следующим набором правил:

	1	a	b	Λ
q_0	$b, q_1, 1$			$q_3, \Lambda, -1$
q_1	$b, q_2, 1$			$q_3, \Lambda, -1$
q_2	$a, q_0, 1$			$q_3, \Lambda, -1$
q_3		$q_4, a, -1$	$q_3, b, -1$	
q_4		$q_4, a, -1$	$q_7, a, 1$	$q_9, \Lambda, 1$
q_5		$q_6, a, -1$	$q_5, b, -1$	
q_6		$q_6, a, -1$	$q_7, a, 1$	$q_8, \Lambda, 1$
q_7		$q_5, b, -1$		
q_8		$q_8, a, 1$	$q_8, b, 1$	$q_3, \Lambda, -1$
q_9		$q_9, a, 1$	$q_9, b, 1$	$q_{10}, \Lambda, -1$
q_{10}		$q_{10}, 1, -1$	$q_{10}, \Lambda, -1$	

«Программа» данной машины разделена на «блоки»: (q_0, q_1, q_2), (q_3, q_4), (q_5, q_6, q_7), (q_8), (q_9) и (q_{10}). Под этим понимается, что каждый блок выполняет свою отдельную задачу, и переход из одного блока означает, что задача текущего блока была выполнена. Под «последовательностью» понимается слово от символа под считывающей головкой в изначальной конфигурации включительно до первого пустого символа справа не включительно.

- **Блок 1: (q_0, q_1, q_2)** — в этом блоке машина «шифрует» входные данные таким образом, что на месте единиц с номерами, кратными 3 (считая, что единица, изначально находившаяся под считывающей головкой, имеет номер 1) стоят символы «a», а на месте остальных — «b». Выполнив этот блок, машина уже никогда к нему не возвращается, то есть при $i > 2$ нет переходов из состояния q_i в состояние q_0, q_1 или q_2. Дойдя до первого пустого символа, машина переходит в Блок 2.

- **Блок 2: (q_3, q_4)** — в этом блоке машина проходит по уже зашифрованной последовательности справа налево и проверяет, верно ли, что все символы «a» стоят левее символов «b». Она начинает с состояния q_3, которое «означает», что справа от положения машины — некоторое количество символов «b», затем — пустой символ. Как только под считывающей головкой оказывается символ «a», машина переходит в состояние q_4, которое «означает», что справа от неё идёт некоторое количество символов «a», затем — некоторое количество символов «b», затем — пустой символ. Как только машина в состоянии q_4 «находит» символ «b», она переходит в Блок 3.

- **Блок 3: (q_5, q_6, q_7)** — в этом блоке машина выполняет одну итерацию сортировки пузырьком над частью последовательности, находящейся слева от начального положения машины при запуске переходе в Блок 3. В этом блоке машина работает так же, как и в Блоке 2 (состоянию q_3 соответствует q_5, а q_4 — q_6), с той разницей, что, встретив в состоянии q_6 символ «b», машина просто меняет его с предыдущим символом (который равен «a», что гарантируется состоянием q_6) и продолжает работу.

- **Блок 4: (q_8)** — в этот блок машина переходит из Блока 3, когда доходит до конца последовательности (до первого пустого символа). Выполняя Блок 4, машина проходит всю последовательность до её правого конца (то есть до первого пустого символа), ничего в ней не меняя, после чего переходит в Блок 2.

- **Блок 5: (q_9)** — в этот блок машина переходит из Блока 2, когда она доходит до конца последовательности (до первого пустого символа). Если это случилось, гарантируется, что последовательность не содержит подпоследовательностей «ba» (иначе машина перешла бы из Блока 2 в Блок 3), то есть последовательность отсортирована. Выполняя Блок 5, машина проходит всю последовательность до её правого конца (то есть до первого пустого символа), ничего в ней не меняя, после чего запускает Блок 6.

- **Блок 6: (q_{10})** — в этом блоке машина идёт по последовательности справа налево и стирает все встречающиеся символы «b», а символы «a» заменяет на «1». При выполнении Блока 6 машина, дойдя до левого конца последовательности (то есть до первого пустого символа), останавливается, так как правила для комбинации (q_{10}, Λ) нет.

Поскольку машина изначально заменяет на символ «a» каждую третью единицу, их число как раз и является неполным частным при делении входного значения на 3. Таким образом, отсортировав последовательность так, что все символы «a» стоят подряд (корректность пузырьковой сортировки можно считать общеизвестной), а затем заменив их на единицы, удалив всё остальное и остановившись слева от полученной унарной последовательности, машина получит как количество символов «a» в зашифрованной исходной последовательности, которое, как было сказано выше, равно искомому неполному частному при делении входного значения на 3.
