Арифметическое кодирование

Дискретный анализ 2012/13

Андрей Калинин, Татьяна Романова

11 мая 2013 г.

Литература

- Witten, Moffat, Bell, Managing gigabytes: compressing and indexing documents and images.
- M. J. Atallah, Algorithms and Theory of Computation Handbook, 12-4 Arithmetic Coding

Раздел

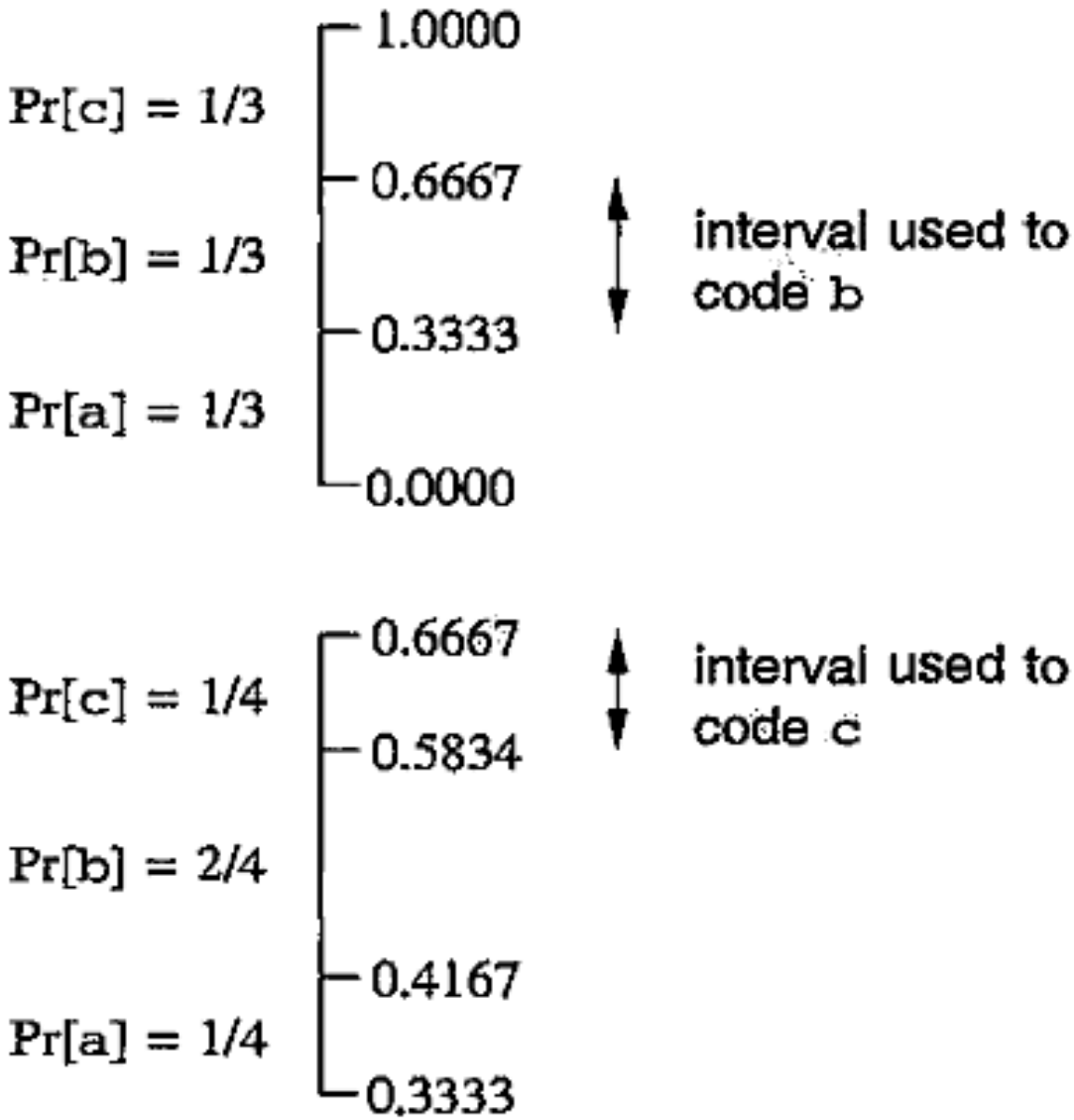
Арифметическое кодирование

- Основная идея
- Реализация
- Дерево Фенвика

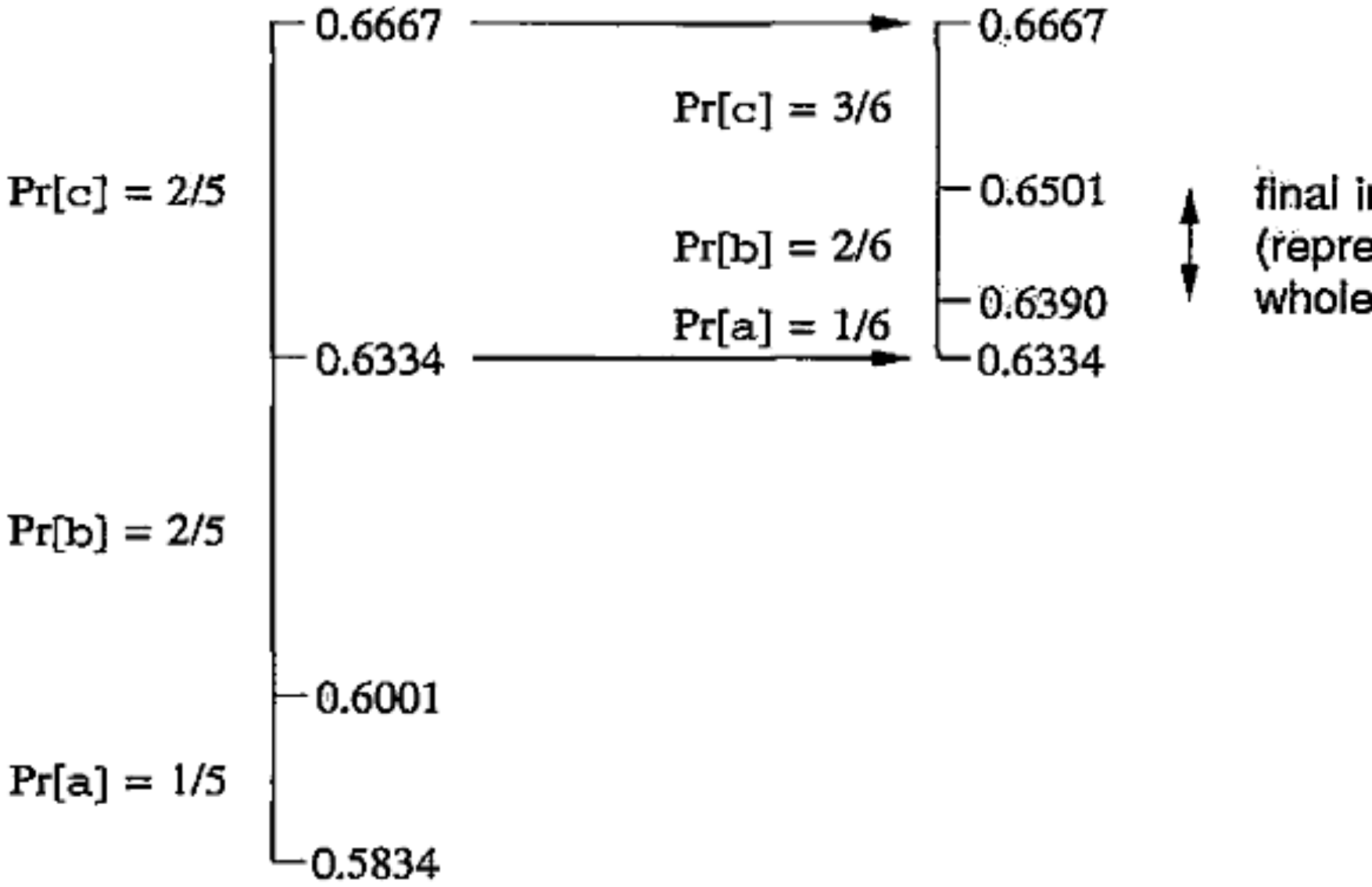
Раздел

Арифметическое кодирование
Основная идея
Реализация
Дерево Фенвика

Кодирование $bccb$



Кодирование $bccb$



Шаг кодирования

1 $low\_bound \leftarrow \sum_{i=1}^{s-1} P[i]$
2 $high\_bound \leftarrow \sum_{i=1}^{s} P[i]$
3 $range \leftarrow high - low$
4 $high \leftarrow low + range \times high\_bound$
5 $low \leftarrow low + range \times low\_bound$

Шаг декодирования

1. Найти s такое, что

$$\sum_{i=1}^{s-1} P[i] \leq \frac{value - low}{high - low} < \sum_{i=1}^{s} P[i]$$

2. Выполнить все шаги по сужению отрезка, аналогичные шагу кодирования.
3. Вернуть символ s.

Раздел

Арифметическое кодирование
- Основная идея
- Реализация
- Дерево Фенвика

Основные моменты

- Вещественные числа имеют конечную точность. Невозможно закодировать гигабайт текста 64-битным вещественным числом.
- Вместо интервала $[0, 1)$ рассмотрим интервал $[0, 2^N - 1)$. Вероятности заменим на количество появлений.
- Если левая и правая границы интервала имеют одинаковый префикс, выведем его и расширим интервал. Выведенные биты дадут декодеру информацию о текущем интервале.
- Если общего префикса нет (середина отерзка между границами), и интервал слишком маленький (меньше, чем кол-во считанных символов), то его тоже нужно расширять, не выводя ничего в поток, но и не теряя информацию (нужна дополнительная переменная-счетчик).

Кодирование. Реализация.

AR-ENCODE-SYMBOL(a_i)

```
1  l ← l + ((h − l + 1) * cum_freq[i])/cum_freq[0]
2  h ← l + ((h − l + 1) * cum_freq[i − 1])/cum_freq[0] − 1
3  repeat
4        if левый бит l = левому биту h
5              AR-SEND-BIT(leftbit(h))
6              l ← 2 * l
7              h ← 2 * h + 1
8        else if h − l < cum_freq[0]
9                    l ← 2 * (l − 2^(N−2))
10                   h ← 2 * (h − 2^(N−2)) + 1
11                   counter = counter + 1
12
13 until левый бит l != левому биту h и h − l ≤ cum_freq[0]
```

Кодирование. Реализация.

AR-SEND-BIT(bit)

```
1   write bit
2   while counter > 0
3         write !bit
4         counter = counter − 1
```

Кодирование. Пример.

Кодируем строчку BBCA. $N = 4, l = 0, h = 15$.

Распределение символов известно заранее:

$f(A) = 1, f(B) = 2, f(C) = 1$.

Массив $cum\_freq = [4, 3, 1, 0]$.

1. Считываем символ B (индекс 2 в массиве cum_freq), изменяем границы:

$$l = 0 + (15 - 0 + 1) * 1/4 = 4 = 0100$$

$$h = 0 + (15 - 0 + 1) * 3/4 - 1 = 11 = 1011$$

Общего префикса нет, вывести ничего нельзя, интервал достаточно большой, идем дальше.

2. Считываем символ B:

$$l = 4 + (11 - 4 + 1) * 1/4 = 6 = 0110$$

$$h = 4 + (11 - 4 + 1) * 3/4 - 1 = 0 = 1001$$

Общего префикса нет, вывести ничего нельзя, интервал слишком короткий.

Кодирование. Пример.

2. Изменяем интервал и увеличиваем $counter$

$$l = (0110 - 0100) << 1 = 0100 = 4$$

$$h = (1001 - 0100) << 1 + 1 = 1011 = 11$$

Общего префикса нет, вывести ничего нельзя, интервал достаточно большой, идем дальше.
3. Считываем символ C:

$$l = 4 + (11 - 4 + 1) * 0/4 = 4 = 0100$$

$$h = 4 + (11 - 4 + 1) * 1/4 - 1 = 5 = 0101$$

Общий префикс 010, после вывода первого бита выводим $counter$ раз его отрицание, а затем все остальное: 0110.
Границы после сдвига: 0000 и 1111

Кодирование. Пример.

4. Считываем A

$$l = 0 + 16 * 3/4 = 12 = 1100$$

$$h = 0 + 16 * 4/4 - 1 = 15 = 1111$$

Общий префикс 11 , выводим.
Итого, на выходе: 011011

Декодирование. Пример.

Последовательность: 011011

При декодировании для изменения интервалов делаем те же шаги, что и при кодировании.

1. Из потока считываем первые N бит: $value = 0110 = 6$, видим, что 6 принадлежит интервалу от 4 до 12, следовательно первая буква B.
2. Изменяем интервал: $l = 4, h = 11$, общих бит нет, $value$ не меняем, оно попадает в интервал от 6 до 19, следовательно вторая буква тоже B.
3. Изменяем интервал $l = 6, h = 9$, он слишком мал, изменяем границы и $value = 2 * (value - 2^(N - 2)) + bit$.
4. Новые границы $l = 4, h = 11, value = 0101 = 5$ => следующая буква C.
5. Изменяем интервал $l = 4, h = 5$ => $value = 1100 = 12$ (биты кончились, дополнили нулями) => следующая буква A.

Раздел

Арифметическое кодирование
Основная идея
Реализация
Дерево Фенвика

Основная идея

- Общие частоты могут быть представлены в виде сумм «подчастот» для поддиапазонов.
- Нужно разделить весь диапазон счётчиков на поддиапазоны, которые можно было бы быстрее обходить, чем все счётчики, начиная с первого.

Индекс	Двоичный	Диапазон	Частота	Сумма	Хранится
0	0000	0	0	0	0
1	0001	1	2	2	2
2	0010	1. . . 2	0	2	2
3	0011	3	1	3	1
4	0100	1. . . 4	1	4	4
5	0101	5	1	5	1
6	0110	5. . . 6	0	5	1
7	0111	7	4	9	4
8	1000	1. . . 8	4	13	13
9	1001	9	0	13	0
10	1010	9. . . 10	1	14	1
11	1011	11	0	14	0
12	1100	9. . . 12	1	15	2
13	1101	13	2	17	2
14	1110	13. . . 14	3	20	5
15	1111	15	0	20	0

Получение суммарной частоты

```
1  S ← Tree[0]
2  while i > 0
3        S ← S + Tree[i]
4        i ← i&(i − 1)
5  return S
```

Изменение частоты

1 **repeat**
2 $\quad Tree[i] \leftarrow Tree[i] + v$
3 $\quad i \leftarrow i + (i\& - i)$
4 **until** $i \geq TableSz$