Б. Я. Рябко, Е. П. Мачикина, Эффективное преобразование случайных последовательностей в равновероятностные и независимые, *Пробл. передачи информ.*, 1999, том 35, выпуск 2, 23–28

ПРОБЛЕМЫ ПЕРЕДАЧИ ИНФОРМАЦИИ

Том 35 **1999** **Вып. 2**

УДК 621.391.15

© 1999 г. **Б.Я. Рябко, Е.П. Мачикина**

ЭФФЕКТИВНОЕ ПРЕОБРАЗОВАНИЕ СЛУЧАЙНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ В РАВНОВЕРОЯТНОСТНЫЕ И НЕЗАВИСИМЫЕ[1]

Решается задача эффективного преобразования последовательностей, порождаемых произвольным бернуллиевским источником, в последовательность независимых и равновероятностных символов, ранее рассматривавшаяся Дж. фон Нейманом, П. Элайесом и другими. У предлагаемого метода, основанного на алгоритме Элайеса, объем памяти и время, затрачиваемое на обработку одного символа, экспоненциально меньше, чем у ранее известных.

§ 1. Введение

В статье решается задача, впервые рассмотренная фон Нейманом [1]: дан бернуллиевский источник, порождающий символы из алфавита $\{0,1\}$ с вероятностями $1-p$ и p соответственно, $0<p<1$, причем p может быть неизвестно. Требуется преобразовать (или закодировать) порождаемую последовательность в алфавите $\{0,1\}$ в такую последовательность, что вероятности появления нуля и единицы равны. (Иногда такую последовательность называют абсолютно случайной.) Эта задача привлекала внимание многих исследователей (см., например, [2, 3]). Фон Нейман предложил следующий метод. Исходная последовательность разбивается на блоки (подслова) длины 2, которые кодируются по следующему правилу:

$$00 \to \Lambda, 01 \to 0, 10 \to 1, 11 \to \Lambda, \tag{1}$$

где Λ обозначает пустое слово [1]. Например, порождаемая последовательность 00011110000001 будет трансформирована в абсолютно случайную последовательность 010. Здесь первый ноль соответствует второму блоку, т.е. 01, единица соответствует четвертому блоку, а второй ноль – последнему блоку.

Так как вероятности порождения слов 01 и 10 совпадают (равны $p(1-p)$ и $(1-p)p$), то, очевидно, в результате преобразования (1) получается абсолютно случайная последовательность.

Из приведенного примера виден и недостаток метода, задаваемого формулой (1), – трансформированная последовательность намного короче исходной.

Элайес [2] предложил метод преобразования, более экономно расходующий символы исходной последовательности, что достигается за счет перехода к кодированию блоков длины N, большей двух. (При $N=2$ методы Элайеса и фон Неймана совпадают.) Для количественной оценки эффективности метода Элайес ввел величину

[1]Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (номер проекта 99-01-000586).

η_N, определяемую как среднее значение отношения длины кодового слова, соответствующего блоку, к длине блока N.

Естественной верхней границей для величины η_N является энтропия источника $H(p)$, определяемая равенством

$$H(p) = -(p\log p + (1-p)\log(1-p)).$$

(Впрочем, понятно, что $H(p)$ – максимально возможное значение отношения длин выходной и входной последовательностей для любого метода, так как энтропия – это мера неопределенности, или случайности, исходной последовательности.)

Дадим теперь описание метода (или кода) Элайеса. Пусть N – длина блока. Разобьем множество 2^N возможных входных слов на $N+1$ классов S_k, $0 \le k \le N$, $S_k = \{(x_1, x_2, \ldots, x_N) \mid x_i \in \{0,1\}, \sum_{i=1}^{N} x_i = k\}$. Другими словами, S_k – множество всех слов длины N с k единицами. Обозначим $m_k = \lfloor \log_2 |S_k| \rfloor = \lfloor \log_2 \binom{N}{k} \rfloor$, где $\lfloor y \rfloor$ – наибольшее целое, не превосходящее y.

Рассмотрим двоичные представления чисел $|S_k| = \binom{N}{k}$, $|S_k| = (\alpha_{m_k}, \alpha_{m_k-1}, \ldots$ $\ldots, \alpha_0)$, причем $\alpha_{m_k} = 1$, $\alpha_j \in \{0,1\}$, $m_k > j \ge 0$. Если $\alpha_j = 1$, $0 \le j \le m_k$, то поставим в однозначное соответствие 2^j различным элементам множества S_k всевозможные слова длины j. Один из элементов S_k будет соответствовать Λ в случае, если $|S_k|$ нечетно. Поскольку S_0 и S_N имеют только по одному элементу каждый, то им, следовательно, будет соответствовать Λ. Полагая $N = 2$, мы получим описанный ранее метод фон Неймана.

Определим избыточность r данного метода как максимум разности $H(p)$ и η_N по всем бернулллиевским источникам:

$$r = \sup_{p \in (0,1)} (H(p) - \eta_N).$$

Элайес показал, что для предложенного им метода $r = O(1/N)$.

Мы будем оценивать сложность метода двумя характеристиками, зависящими от избыточности r (или длины блока N): объемом используемой памяти (в битах) и временем, затрачиваемым на кодирование одной буквы исходной последовательности, измеряемым в операциях над однобитовыми словами. Для определенности будем считать, что методы реализованы в виде программ на компьютере со свободным доступом к памяти, являющемся моделью "обычного" компьютера [4].

Как мы видим из описания метода Элайеса, он требует хранения всех 2^N кодовых слов, поэтому размер памяти кодера возрастает экспоненциально с ростом N. Ниже мы предлагаем быстрый метод кодирования для кода Элайеса, при котором требуется неэкспоненциально растущий размер памяти. Предлагаемый алгоритм основан на методе нумерационного кодирования из [5, 6] и использует алгоритм быстрого умножения Шенхаге–Штрассена.

§ 2. Быстрый метод кодирования

Пусть $x_1, x_2, \ldots, x_n, \ldots$ – последовательность двоичных символов, порождаемых бернуллиевским источником с вероятностями $\Pr\{x_n = 1\} = p$, $\Pr\{x_n = 0\} = 1 - p$, и пусть $N \ge 2$ – длина блока. Опишем кодирование произвольного слова x^N длины N по предлагаемому алгоритму.

Пусть x^N содержит k единиц, т.е. $x^N \in S_k$, $k = 0, 1, \ldots, N$. Упорядочим слова из S_k лексикографически, и пусть $\mathrm{Num}(x^N)$ – номер x^N в S_k при таком упорядочении. Разобьем описание кодирования на три этапа:

1. Вычислим номер $\mathrm{Num}(x^N)$;

2. Выпишем двоичное представление числа $|S_k|$: $|S_k| = (\alpha_{m_k}, \ldots, \alpha_j, \ldots, \alpha_i, \ldots \ldots, \alpha_0)$. Найдем пару индексов i, j, $0 \le i < j \le m_k$, которые удовлетворяют условиям

$$\alpha_i = 1, \quad \alpha_j = 1, \quad \alpha_t = 0, \quad t = i+1, \ldots, j-1,$$
$$\sum_{r=0}^{i} \alpha_r 2^r \le \mathrm{Num}(x^N) < \sum_{r=0}^{i} \alpha_r 2^r + 2^j; \tag{2}$$

3. По определению кодовое слово $\mathrm{code}(x^N)$ задается j младшими двоичными символами $\mathrm{Num}(x^N)$.

Как ясно из описания, самый трудоемкий этап кодирования – вычисление $\mathrm{Num}(x^N)$. Для описания "быстрого" алгоритма мы сначала опишем метод нумерации слов из S_k, предложенный в [7–10]. (Он был независимо открыт Бабкиным, Линчем, Девиссоном.) В этих работах предлагается метод, основанный на равенстве

$$\mathrm{Num}(x^N) = \sum_{t=1}^{N} x_t \binom{N-t}{k - \sum_{i=1}^{t-1} x_i}. \tag{3}$$

Можно легко показать, что вычисления по формуле (3) требуют $\mathrm{const}N^2$ битовых операций, или $\mathrm{const}N$ операций на букву.

Предлагаемый метод расходует только $O(\log^3 N \log\log N)$ операций на букву. Он строится аналогично коду из [5, 6].

Для описания этого метода сначала представим (3) в виде

$$\mathrm{Num}(x^N) = \binom{N}{k} \left(x_1 \frac{\binom{N-1}{k}}{\binom{N}{k}} + x_2 \frac{\binom{N-2}{k - \sum_{i=1}^{1} x_i}}{\binom{N-1}{k - \sum_{i=1}^{1} x_i}} \frac{\binom{N-1}{k - \sum_{i=1}^{1} x_i}}{\binom{N}{k}} + \ldots \right).$$

Введем вспомогательные величины (как это сделано в [5, 6]). Пусть

$$p(x_t/x_1, \ldots, x_{t-1}) = \frac{\binom{N-t}{k - \sum_{i=1}^{t} x_i}}{\binom{N-t+1}{k - \sum_{i=1}^{t-1} x_i}} = x_t Q_t + (1 - x_t)(1 - Q_t),$$

$$q(x_t/x_1, \ldots, x_{t-1}) = x_t \frac{\binom{N-t}{k - \sum_{i=1}^{t-1} x_i}}{\binom{N-t+1}{k - \sum_{i=1}^{t-1} x_i}} = x_t(1 - Q_t),$$

где $Q_t = \dfrac{k - \sum_{j=1}^{t-1} x_j}{N - t + 1}$, $t = 1, \ldots, N$.

Тогда

$$\mathrm{Num}(x_1, x_2, \ldots, x_N) =$$
$$= |S_k|(q(x_1) + q(x_2/x_1)p(x_1) + q(x_3/x_2, x_1)p(x_2/x_1)p(x_1) + \ldots). \qquad (4)$$

Для вычисления $\mathrm{Num}(x^N)$ мы используем быстрый нумерационный код, предложенный в [5, 6]. Для простоты можно считать, что $\log N$ является целым числом, в противном случае мы можем добавить символ 0 к каждому слову из S_k, чтобы сделать $\log N$ целым, причем мощность S_k и сложность кода не изменятся. Теперь определим величины

$$\rho_1^0 = p(x_1),\ \rho_2^0 = p(x_2/x_1),\ \ldots,\ \rho_N^0 = p(x_N/x_1, \ldots, x_{N-1}), \qquad (5)$$
$$\lambda_1^0 = q(x_1),\ \lambda_2^0 = q(x_2/x_1),\ \ldots,\ \lambda_N^0 = q(x_N/x_1, \ldots, x_{N-1}), \qquad (6)$$

$$\rho_t^s = \rho_{2t-1}^{s-1}\rho_{2t}^{s-1},\ \lambda_t^s = \lambda_{2t-1}^{s-1} + \lambda_{2t}^{s-1}\rho_{2t}^{s-1},\ s = 1, 2, \ldots, \log N,\ t = 1, \ldots, N/2^s.$$

Нетрудно видеть, что

$$\mathrm{Num}(x^N) = |S_k|\lambda_1^{\log N}. \qquad (7)$$

Величина $\mathrm{Num}(x^N)$ вычисляется в соответствии с (6), (7), причем все величины представляются рациональными числами, задаваемыми числителем и знаменателем.

Приведем пример вычисления по формулам (5), (6). Пусть $N = 4$, $k = 2$, и $x^N = (1001)$. Тогда

$$\rho_1^0 = p(x_1) = Q_1 = \frac{2}{4-1+1} = \frac{1}{2},$$
$$\rho_2^0 = p(x_2/x_1) = 1 - Q_2 = 1 - \frac{2-1}{4-2+1} = \frac{2}{3},$$
$$\rho_3^0 = p(x_3/x_1, x_2) = 1 - Q_3 = 1 - \frac{2-1}{4-3+1} = \frac{1}{2},$$
$$\rho_4^0 = p(x_4/x_1, x_2, x_3) = Q_4 = \frac{2-1}{4-4+1} = 1,$$
$$\lambda_1^0 = q(x_1) = 1 - Q_1 = \frac{1}{2},$$
$$\lambda_2^0 = q(x_2/x_1) = 0,$$
$$\lambda_3^0 = q(x_3/x_1, x_2) = 0,$$
$$\lambda_4^0 = q(x_4/x_1, x_2, x_3) = 0,$$
$$\rho_1^1 = \rho_1^0\rho_2^0 = \frac{1}{2}\cdot\frac{2}{3} = \frac{1}{3},$$
$$\rho_2^1 = \rho_3^0\rho_3^0 = \frac{1}{2}\cdot 1 = \frac{1}{2},$$
$$\lambda_1^1 = \lambda_1^0 + \lambda_2^0\rho_2^0 = \frac{1}{2},$$
$$\lambda_2^1 = \lambda_3^0 + \lambda_4^0\rho_4^0 = 0,$$
$$\lambda_2^2 = \lambda_1^1 + \lambda_2^1\rho_2^1 = \frac{1}{2}.$$

Используя (7), мы получим $\mathrm{Num}(1001) = |S_2|\lambda_2^2 = 6\cdot\frac{1}{2} = 3$, что, конечно, является номером слова (1001) в S_2 при лексикографическом упорядочивании. Свойства описанного метода характеризует следующая

Теорема. *Пусть* $x_1, x_2, \ldots, x_n, \ldots$ *является последовательностью двоичных символов, порожденной бернуллиевским источником с вероятностями* $\Pr\{x_n = 1\} = p$, $\Pr\{x_n = 0\} = 1 - p$, $0 < p < 1$. *Предложенный метод для преобразования входной последовательности в последовательность независимых равновероятностных символов имеет следующие свойства:*

1) *избыточность* $r = O(\frac{1}{N})$,

2) *время кодирования одного символа* $T = O(\log^3 N \log\log N)$ *битовых операций,*

3) *требуемый объем памяти* $M = O(N \log^2 N)$ *бит,*

где N *– длина блока.*

Доказательство. Очевидно, что избыточность метода Элайеса совпадает с избыточностью предложенного выше метода. Отсюда немедленно следует утверждение 1).

Для упрощения обозначений будем считать, что $\log N$ является целым числом. Все вычисления проводятся с рациональными числами; все ρ_t^s и λ_t^s являются дробями и представлены парой целых чисел. Для умножения используется метод Шенхаге–Штрассена как наиболее быстрый [4]. Для этого метода время умножения $t(L)$ двух двоичных чисел длины L (и время деления $(2L)$-разрядного числа на L-разрядное) составляет

$$t(L) = O(L \log L \log\log L), \quad L \to \infty. \tag{8}$$

Нетрудно видеть, что двоичное представление для каждого $p(x_t/x_1, \ldots, x_{t-1})$ и $q(x_t/x_1, \ldots, x_{t-1})$ использует $2\log N$ разрядов ($\log N$ – для числителя, и $\log N$ – для знаменателя). Поэтому вычисление ρ_t^1, $t = 1, \ldots, N/2$, по формуле (6) требует $2(N/2)$ умножений чисел с $\log N$ разрядами, и вычисление λ_t^1, $t = 1, \ldots, N/2$, согласно формуле (6) и тождеству $a/b + c/d = (ad + bc)/(bd)$ требует $3(N/2)$ умножений ($\log N$)-разрядных чисел. Вычисление ρ_t^2, λ_t^2, $t = 1, \ldots, N/4$, требует $5(N/4)$ умножений чисел с $2\log N$ разрядами. Аналогично получаем, что для вычисления ρ_t^i, λ_t^i, $t = 1, \ldots, N/2^i$, требуется $5(N/2^i)$ умножений чисел с $2^i \log N$ разрядами. Используя (8), получаем, что время вычисления составляет

$$\begin{aligned}
&(5N/2)O(\log N \log\log N \log\log\log N) + (5N/4)O(2\log N \log(2\log N)\log\log(2\log N)) + \ldots \\
&\ldots + (5N/2^i)O(2^i \log N \log(2^i \log N)\log\log(2^i \log N)) + \ldots \\
&\ldots + 5O(N \log N \log(N \log N)\log\log(N \log N))
\end{aligned}$$

битовых операций, и оно не больше, чем

$$O(N \log^2 N \log(N \log N)\log\log(N \log N)).$$

Это дает $O(\log^3 N \log\log N)$ битовых операций на один входной символ для вычисления $\lambda_1^{\log N}$. Чтобы получить $\mathrm{Num}(x^N)$, мы должны согласно (7) вычислить $|S_k|\lambda_1^{\log N}$. Очевидно, что $|S_k| < 2^N$, и двоичное представление имеет N разрядов. Длина $\lambda_1^{\log N}$ составляет $N \log N$ разрядов. Из (8) получаем, что время вычисления $|S_k|\lambda_1^{\log N}$ требует $O(\log^2 N \log\log N)$ битовых операций на один входной символ. Нахождение кодового слова $\mathrm{code}(x^N)$ по номеру $\mathrm{Num}(x^N)$ требует не более $O(1)$ битовых операций на один входной символ. Окончательно мы имеем утверждение 2).

Для оценки требуемого объема памяти заметим, что при вычислении ρ_k^i, λ_k^i необходимо хранить ρ_k^{i-1}, λ_k^{i-1}, $i = 2, \ldots, \log N$, и что одинаковая память требуется для хранения $\{\rho_k^{i-1}, \lambda_k^{i-1}, k = 1, \ldots, N/2^{i-1}\}$ и $\{\rho_k^i, \lambda_k^i, k = 1, \ldots, N/2^i\}$. Объем памяти, требуемой для вычисления кодового слова $\mathrm{code}(x^N)$ по $\mathrm{Num}(x^N)$, составляет $O(N)$ бит (необходимо хранить двоичное представление $|S_k| = \binom{N}{k}$). Отсюда получаем утверждение 3).

Рассмотрим пример построения абсолютно случайной последовательности из данной входной последовательности. Пусть дана входная последовательность 101011110110. Будем кодировать блоки длины $N = 4$. Сначала вычислим $\mathrm{Num}(1010)$ в множестве S_2. Согласно (4), (6) мы получим, что $\mathrm{Num}(1010) = 4 = (100)_2$. Двоичное представление числа $|S_2| = (110)_2$. Поскольку $2^1 \leq \mathrm{Num}(1010) < 2^1 + 2^2$, то в качестве кодового слова следует взять два младших двоичных разряда, т.е. $\mathrm{code}(1010) = (00)$. Аналогично получим, что $\mathrm{code}(1111) = \Lambda$, $\mathrm{code}(0110) = (11)$. Окончательно имеем выходную последовательность 0011.

СПИСОК ЛИТЕРАТУРЫ

1. *von Neumann J.* Various Techniques Used in Connection with Random Digits // Monte Carlo Method, Applied Mathematics. Ser. 12. Washington: U.S. National Bureau of Standarts, 1952. P. 36–38.
2. *Elias P.* The Efficient Construction of an Unbiased Random Sequence // Ann. Math. Statist. 1972. V. 43. № 3. P. 864–870.
3. *Vembu S., Verdú S.* Generating Random Bits from an Arbitrary Source: Fundamental Limits // IEEE Trans. Inform. Theory. 1995. V. 41. № 5. P. 1322–1332.
4. *Ахо А., Хопкрофт Дж., Ульман Дж.* Построение и анализ вычислительных алгоритмов. М.: Мир, 1979.
5. *Ryabko B.Ya.* Fast and Efficient Coding of Information Sources // IEEE Trans. Inform. Theory. 1994. V. 40. № 1. P. 96–99.
6. *Рябко Б.Я.* Быстрая нумерация комбинаторных объектов // Дискретная математика. 1998. Т. 10. № 2. С. 101–110.
7. *Cover T.M.* Enumerative Source Encoding // IEEE Trans. Inform. Theory. 1973. V. 19. № 1. P. 73–77.
8. *Бабкин В.Ф.* Метод универсального кодирования источника независимых сообщений неэкспоненциальной трудоемкости // Пробл. передачи информ. 1971. Т.7. № 4. С. 13–21.
9. *Lynch T.Y.* Sequence Time Coding for Data Compression // Proc. IEEE. 1966. V. 54. № 1. P. 1490–1491.
10. *Davisson L.D.* Comments on “Sequence Time Coding for Data Compression” // Proc. IEEE. 1966. V. 54. № 2. P. 2010.

Поступила в редакцию
03.04.98
После переработки
13.10.98