УДК 621.391.1-681.3

# АДАПТИВНОЕ АРИФМЕТИЧЕСКОЕ КОДИРОВАНИЕ В СТАНДАРТЕ JPEG 2000

***Е. А. Беляев,***
*аспирант*
***А. М. Тюрликов,***
*канд. техн. наук, доцент*
***А. С. Уханова,***
*студентка*
*Санкт-Петербургский государственный университет аэрокосмического приборостроения*

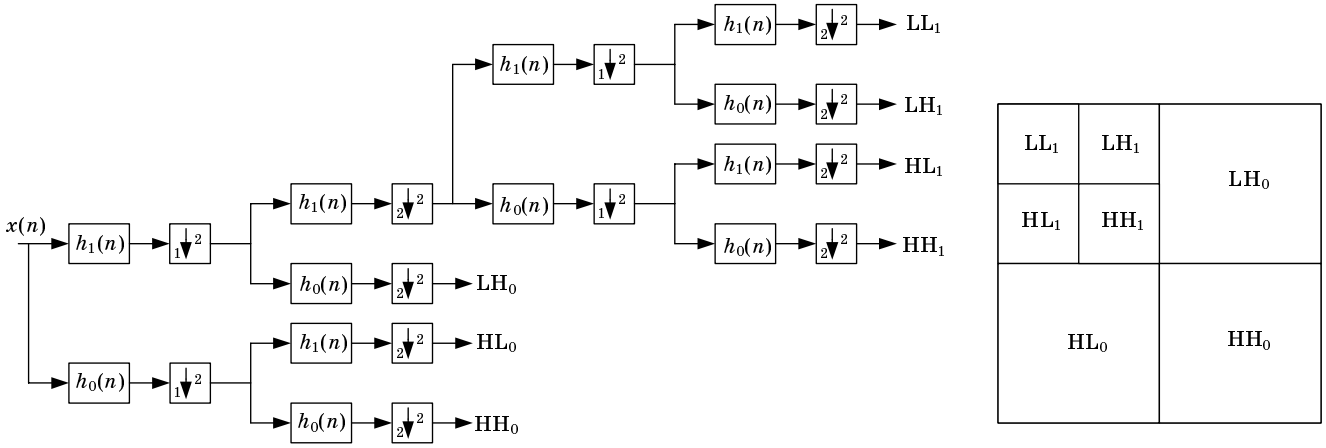
*Представлен обзор контекстного моделирования, используемого в стандарте JPEG 2000. Описан способ модификации механизма адаптивной оценки вероятности появления символа, позволяющий учесть статистические особенности двоичных источников, соответствующих различным контекстным моделям, и тем самым повысить эффективность алгоритма.*

*This is a review of context modelling used in the JPEG 2000 standard. We show how to modify the adaptive estimation mechanism of the probability of symbol appearance, taking into account the statistical peculiarities of binary sources corresponding to different context models and thus improving the algorithm.*

## Введение

В последнее время стандарт JPEG 2000 [1–3] все более распространяется в качестве основного формата сжатия как изображений, так и видеопоследовательностей.

Схема кодирования, используемая в стандарте JPEG 2000, кратко может быть описана следующим образом. Исходное изображение разбивается на прямоугольные сегменты (tile), которые кодируются независимо. После преобразования цветового пространства каждый сегмент изображения подвергается процедуре многоуровневого дискретного вейвлетного преобразования (discrete wavelet transform). Для этого используются низкочастотный $h_0(n)$ и высокочастотный $h_1(n)$ вейвлетные фильтры. При разложении сигнала сегмента изображения вначале выполняется разложение по строкам, а затем по столбцам. Результатом разложения являются 4 матрицы: $HH_0$, $HL_0$, $LH_0$, $LL_0$, соответствующие фильтрации фильтром $h_1(n)$ по строкам и по столбцам, фильтром $h_1(n)$ по стро-




■ ***Рис. 1.*** *Вейвлетное разложение изображения*

кам и $h_0(n)$ по столбцам, фильтром $h_0(n)$ по строкам и $h_1(n)$ по столбцам, фильтром $h_0(n)$ по строкам и столбцам. Далее производится децимация (прореживание) полученных матриц по строкам и столбцам с коэффициентом 2. Затем матрица $LL_0$ подвергается дальнейшему вейвлетному разложению [4]. Его результатом являются матрицы $HH_1$, $HL_1$, $LH_1$, $LL_1$ (рис. 1). Такое разложение повторяется $v$ раз. Результатом разложения является набор из $3v + 1$ матриц уменьшающейся размерности.

Каждая из матриц разложения делится на блоки, которые кодируются независимо друг от друга при помощи контекстного адаптивного двоичного арифметического кодера.

### Арифметическое кодирование и контекстное моделирование

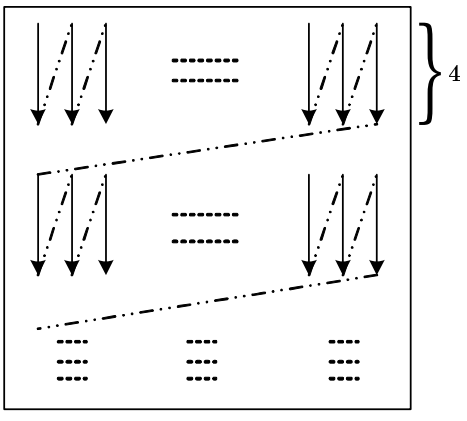
Коэффициенты матриц вейвлетного разложения представляются в двоичном виде. Все биты коэффициентов, относящихся к одному разряду, образуют битовую плоскость. Нумерация битовых плоскостей начинается с самого старшего разряда. Самая старшая битовая плоскость, в которой есть хотя бы один ненулевой коэффициент, является наиболее значимой.

Кодирование коэффициентов матриц разложения происходит на уровне битовых плоскостей, при этом каждая битовая плоскость кодируется с помощью арифметического кодера за три прохода: первый называется significance propagation, второй — magnitude refinement и третий — clean-up.

Кодирование плоскостей начинается с наиболее значимой битовой плоскости, для которой применяется только третий проход (clean-up), для всех остальных битовых плоскостей применяются все три прохода.

Битовые плоскости сканируются в порядке, показанном на рис. 2.

Каждому коэффициенту в блоке на этапе кодирования соответствует степень значимости (significance state). Степень значимости может принимать два значения: 1, т. е. коэффициент считается значимым, и 0, что говорит о том, что коэффициент незначимый. Первоначально все коэффициенты считаются незначимыми. После кодирования первого значимого бита (т. е. единицы) коэффициент становится значимым.




■ ***Рис. 2.*** *Порядок сканирования битовых плоскостей*

| | | |
|---|---|---|
| $D_0$ | $V_0$ | $D_1$ |
| $H_0$ | $X$ | $H_1$ |
| $D_2$ | $V_1$ | $D_3$ |

■ ***Рис. 3.*** *Формирование контекста для коэффициента X*

Каждому коэффициенту сопоставляется 8-разрядный «контекстный вектор», состоящий из степеней значимости восьми соседних коэффициентов (рис. 3). По сумме компонент вектора определяется номер контекста.

**Первый проход (significance propagation).** Данный проход включает в себя кодирование бит незначимых коэффициентов, которые имеют ненулевой контекст (табл. 1).

Если на данном проходе кодируется единичный бит, то степень значимости коэффициента, к которому этот бит относится, становится равной 1, после чего кодируется знак коэффициента.

*Кодирование знака.* Номер контекста при кодировании знака зависит от знака и значимости соседних коэффициентов по вертикали ($V_0$, $V_1$) и по горизонтали ($H_0$, $H_1$) (табл. 2, 3). Для указания зависимостей введем следующую функцию от двух аргументов: $f(X_0, X_1)$. Каждый аргумент представляет собой вектор $X = \{S, \text{Sign}\}$, где $S \in \{0, 1\}$, $\text{Sign} \in \{+, -\}$.

После вычисления контекста кодируемый знак представляется как сумма по модулю 2 между действительным значением и предсказанным.

**Второй проход (magnitude refinement).** На данном проходе кодируются биты значимых коэффициентов. Номер контекста вычисляется по всем соседним коэффициентам. При этом делается различие между битом, стоящим сразу после первого значимого бита, и оставшимися битами этого коэффициента (табл. 4).

**Третий проход (clean-up).** На этом проходе кодируется наиболее значимая битовая плоскость, а также биты незначимых коэффициентов, для которых значение контекста во время первого прохода было равно 0. На этом проходе используются контексты первого прохода (табл. 1), а также еще два специальных контекста.

Если в столбце (см. рис. 1) четыре идущих подряд коэффициента кодируются на этом проходе и значения контекстов для них равны 0, тогда используется специальный контекст «Run-length». Если все коэффициенты остаются незначимыми (все 4 бита в столбце в текущей битовой плоскости

■ ***Таблица 1.*** *Контексты для significance propagation*

| LL и LH | | | HL | | | HH | | Номер контекста |
|---|---|---|---|---|---|---|---|---|
| $\sum H$ | $\sum V$ | $\sum D$ | $\sum H$ | $\sum V$ | $\sum D$ | $\sum(H+V)$ | $\sum D$ | |
| 2 | * | * | * | 2 | * | * | ≥ 3 | 8 |
| 1 | ≥ 1 | * | ≥ 1 | 1 | * | ≥ 1 | 2 | 7 |
| 1 | 0 | ≥ 1 | 0 | 1 | ≥ 1 | 0 | 2 | 6 |
| 1 | 0 | 0 | 0 | 1 | 0 | ≥ 2 | 1 | 5 |
| 0 | 2 | * | 2 | 0 | * | 1 | 1 | 4 |
| 0 | 1 | * | 1 | 0 | * | 0 | 1 | 3 |
| 0 | 0 | ≥ 2 | 0 | 0 | ≥ 2 | ≥ 2 | 0 | 2 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

* — любое значение.

■ ***Таблица 2.*** *Вычисление значений функции $f(X_0, X_1)$ по соседним коэффициентам*

| $X_0$ | | $X_1$ | | $f(X_0, X_1)$ |
|---|---|---|---|---|
| Значимость $S$ | Знак Sign | Значимость $S$ | Знак Sign | |
| 1 | + | 1 | + | 1 |
| 1 | – | 1 | + | 0 |
| 0 | * | 1 | + | 1 |
| 1 | + | 1 | – | 0 |
| 1 | – | 1 | – | –1 |
| 0 | * | 1 | – | –1 |
| 1 | + | 0 | * | 1 |
| 1 | – | 0 | * | –1 |
| 0 | * | 0 | * | 1 |

* — любое значение.

■ ***Таблица 3.*** *Вычисление контекста при кодировании знака*

| $f(H_0, H_1)$ для $H$ | $f(V_0, V_1)$ для $V$ | Номер контекста | Предсказанный знак* |
|---|---|---|---|
| 1 | 1 | 13 | 0 |
| 1 | 0 | 12 | 0 |
| 1 | –1 | 11 | 0 |
| 0 | 1 | 10 | 0 |
| 0 | 0 | 9 | 0 |
| 0 | –1 | 10 | 1 |
| –1 | 1 | 11 | 1 |
| –1 | 0 | 12 | 1 |
| –1 | –1 | 13 | 1 |

* 0 — положительное значение, 1 — отрицательное значение.

■ ***Таблица 4.*** *Контексты для magnitude refinement*

| $\sum H + \sum V + \sum D$ | Является ли бит первым после первого значимого бита? | Номер контекста |
|---|---|---|
| – | Нет | 16 |
| ≥ 1 | Да | 15 |
| 0 | Да | 14 |

равны 0), то вместе с контекстом «Run-length» на арифметический кодер подается 0. Если хотя бы один из коэффициентов становится значимым на этом проходе (значение бита на текущей битовой плоскости равно 1), то на арифметический кодер подается 1, а далее кодируются еще 2 бита, которые сообщают о местоположении первой единицы в колонке. Так как вероятности этих двух бит равномерно распределены, то для их кодирования используется еще один специальный контекст «Uniform».

После того как позиция первого ненулевого бита определена, оставшиеся значения в колонке ко-

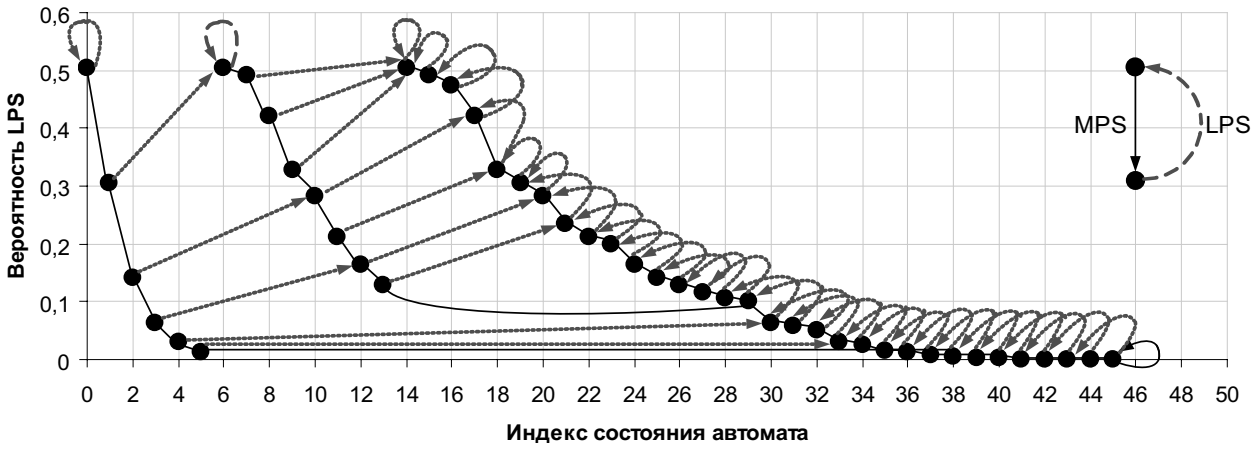
дируются так же, как на первом проходе с использованием тех же 9 контекстов.

### Адаптивная оценка вероятности в арифметическом кодере

В стандарте JPEG 2000 используется 19 контекстных моделей. Для кодирования двоичных источников, соответствующих этим контекстным моделям, используется MQ-coder [1], в котором оценка вероятности появления символа на выходе источника вычисляется при помощи конечного автомата, заданного в виде таблицы состояний и переходов. Входные символы делятся на наиболее вероятные символы (Most Probable Symbol – MPS) и наименее вероятные символы (Least Probable Symbol – LPS). Каждое состояние автомата соответствует оценке вероятности наименее вероятного символа. В зависимости от значения входного символа $x(t)$ (MPS или LPS) происходит переход из одного состояния автомата в другое. Подобная реализация не требует операций умножения/деления как при вычислении оценок вероятности, так и при непосредственно арифметическом кодировании. Пример работы конечного автомата, используемого в MQ-coder, показан на рис. 4. Здесь фактически реализованы три схемы адаптивной оценки, отличающиеся скоростью адаптации. Перед кодированием блока каждой контекстной модели сопоставляется начальное состояние автомата. Такая процедура называется *инициализацией контекстов*.

В табл. 5 приведены инициализационные значения состояний автомата для контекстных моделей, используемых в стандарте. Из таблицы следует, что большинство контекстных моделей инициализируются состоянием «0». Это означает, что после инициализации оценка вероятности, соответствующая этим контекстным моделям, принимает значение, близкое к 0,5, и в начальные моменты времени используется схема с высокой скоростью адаптации. После кодирования некоторого количества двоичных символов происходит переход к схеме с более низкой скоростью адаптации, но с более высокой точностью оценки вероятности.

Описанная выше схема адаптивной оценки вероятности появления символа обладает высокой степенью эффективности, так как обеспечивает быструю адаптацию в начальные моменты времени и более точную оценку вероятности в последующие моменты. Однако эта схема не учитывает того, что статистические свойства двоичных источников, соответствующих различным контекстным моделям, могут существенно отличаться. Ниже описан подход, позволяющий частично учесть эти статистические отличия.




■ ***Рис. 4.*** *Оценка вероятности в MQ-coder*

■ ***Таблица 5.*** *Инициализация контекстов*

| Контекст | Индекс состояния автомата | Вероятность LPS |
|---|---|---|
| «Uniform» | 46 | 0,503937 |
| «Run-length» | 3 | 0,063012 |
| 0 | 4 | 0,030053 |
| Все остальные | 0 | 0,503937 |

### Адаптивная оценка вероятности при помощи «виртуального скользящего окна»

В работах [5, 6] предложен алгоритм адаптивной оценки вероятности появления символа при помощи «виртуального скользящего окна». Оценка вероятности появления единицы для алгоритма «виртуального скользящего окна» определяется как

$$\hat{p}(t) = \frac{s(t)}{W^2},$$

где $W$ — длина «виртуального скользящего окна»; $s(t)$ — состояние конечного автомата, вычисляемое следующим образом:

$$s(t+1) = s(t) + \left\lfloor \frac{W^2 - s(t) + \frac{W}{2}}{W} \right\rfloor \text{ при } x(t) = 1;$$

$$s(t+1) = s(t) - \left\lfloor \frac{s(t) + \frac{W}{2}}{W} \right\rfloor \text{ при } x(t) = 0.$$

Таким образом, каждой контекстной модели можно сопоставить значение параметра $W$, наилучшим образом учитывающего статистические особенности источника. Кроме того, применение алгоритма «виртуального скользящего окна» позволяет отказаться от использования таблицы состояний и переходов. При выборе $W = 2^i$, где $i$ — положительное целое число, вместо операции деления можно использовать операцию побитового сдвига.

### Практические результаты

Алгоритмы сжатия видеоинформации сравниваются по двум основным параметрам: визуальному качеству декодированной видеопоследовательности и битовой скорости (число бит на выходе кодера в единицу времени). Для получения практических результатов алгоритм оценки вероятности, используемый в кодере и декодере JPEG 2000, был заменен алгоритмом «виртуального скользящего окна». Для этого использовалась открытая реализация стандарта JASPER, версии 1.701.0.

Результаты были получены для первого кадра известных тестовых HDTV-видеопоследовательностей («riverbed», «rush hour», «station», «sunflower», «tractor») разрешением 1920×1080. Перед запуском кодера/декодера каждой контекстной модели $i \in \{0, 1, \ldots, 18\}$ сопоставлялась оптимальная длина окна $W_{\text{opt}}(i)$, которая выбиралась следующим образом. В процессе кодирования тестовой видеопоследовательности (в данном случае использовалась последовательность «tractor») для каждого двоичного символа с номером $j$ источника, соответствующего контекстной модели с номером $i$, и для каждой длины окна $W_l = 2^l$ вычислялись оценки вероятности кодируемого символа $\hat{p}_j(i, W_l)$. По завершении кодирования вычислялись оценки битовых затрат

$$\hat{R}(W_l, i) = -\sum_j \log_2 \hat{p}_j(i, W_l), \quad l = 3, 4, \ldots, 10.$$

Для контекстной модели с номером $i$ параметр кодирования устанавливался

$$W_{\text{opt}}(i) = \arg \min_{W_l} \hat{R}(W_l, i).$$

В представленных ниже таблицах приведено уменьшение битовой скорости (в процентах) относительно оригинальной версии кодера при сжатии без потери качества, в зависимости от размера сегмента, для которого выполняется вейвлет-преобразование. Табл. 6, 7 относятся к случаю инициализации контекстов перед кодированием каждого блока каждого сегмента соответственно.

Как видно из табл. 6, 7, сопоставление каждому двоичному источнику параметра кодирования, учитывающего его статистические особенности, приводит к увеличению сжатия. При этом если инициализация контекстов выполняется значительно реже (табл. 7), то эффективность использования предложенной схемы адаптации повышается. Однако такой способ инициализации возможен только для случая сжатия без потери качества.

■ ***Таблица 6**. Уменьшение битовой скорости при использовании алгоритма «виртуального скользящего окна» при инициализации контекстов перед кодированием каждого блока*

| Размер tile | riverbed | rush hour | station | sunflower | tractor |
|---|---|---|---|---|---|
| 1920×16 | 0,76 | 0,76 | 0,74 | 0,84 | 0,85 |
| 1920×32 | 0,44 | 0,31 | 0,36 | 0,31 | 0,57 |
| 1920×64 | 0,39 | 0,20 | 0,32 | 0,14 | 0,56 |
| 1920×128 | 0,58 | 0,38 | 0,49 | 0,30 | 0,68 |
| 1920×256 | 0,68 | 0,45 | 0,58 | 0,40 | 0,76 |
| 1920×1080 | 0,73 | 0,48 | 0,60 | 0,42 | 0,77 |

■ ***Таблица 7.*** *Уменьшение битовой скорости при использовании алгоритма «виртуального скользящего окна» при инициализации контекстов перед кодированием каждого сегмента*

| Размер tile | riverbed | rush hour | station | sunflower | tractor |
|---|---|---|---|---|---|
| 1920×16 | 1,74 | 1,79 | 1,68 | 1,80 | 1,73 |
| 1920×32 | 0,93 | 0,85 | 0,86 | 0,81 | 1,04 |
| 1920×64 | 0,71 | 0,58 | 0,64 | 0,51 | 0,84 |
| 1920×128 | 0,75 | 0,57 | 0,64 | 0,44 | 0,86 |
| 1920×256 | 0,83 | 0,64 | 0,69 | 0,52 | 0,91 |
| 1920×1080 | 0,85 | 0,67 | 0,72 | 0,53 | 0,92 |

В отличие от схемы адаптации, входящей в MQ-coder, алгоритм «виртуального скользящего окна» может быть реализован без использования таблицы состояния и переходов. При этом он является более предпочтительным с точки зрения эффективности кодирования.

## Литература


1. ITU-T and ISO/IEC JTC 1, JPEG 2000 Image Coding System: Core Coding System, ITU-T Recommendation T.800 and ISO/IEC 15444-1 (JPEG 2000 Part 1), 2000.
2. **Rabbani M., Joshi R.** An overview of the JPEG 2000 still image compression standard: Signal Processing: Image Communication 17. 2002.
3. **Adams M. D.** The JPEG-2000 still image compression standard. ISO/IEC JTC 1/SC 29/WG 1 N2412. September 2001. http://www.ece.uvic.ca/mdadams and distributed with the JasPer software.
4. **Mallat S.** A Theory of Multiresolution Signal Decomposition: The Wavelet Representation // IEEE Transactions on Pattern Anal. Mach. Intell. 1989. Vol. 11. P. 674–693.
5. **Belyaev E., Gilmutdinov M., Turlikov A.** Binary Arithmetic Coding System with Adaptive Probability Estimation // Virtual Sliding Window: Proc. of the 10th IEEE International Symposium on Consumer Electronics – ISCE'06. St.-Petersburg, Russia, 2006. P. 194–198.
6. **Беляев Е. А.** Использование «виртуального скользящего окна» при адаптивном арифметическом кодировании// Научная сессия ГУАП, посвященная Всемирному Дню авиации и космонавтики и 65-летию ГУАП: Сб. докл./ ГУАП. СПб., 2006.