

поиска, соответствующую данному решению, будем называть α_m -ветвью. Следующая вершина α_m -ветви помечается переменной x_{m-1} . Данная вершина является корнем дерева поиска для системы неравенств, полученной из (1) подстановкой $x_m = \alpha_m$. Если хотя бы одно из неравенств в (1) при этом не выполняется, то рассматриваемая ветвь отсекается (поскольку заведомо не содержит решения, дающего меньшую верхнюю границу, чем текущая). Описанный процесс по аналогии продолжается далее. Если найден вектор со значением нормы, меньшим текущей верхней границы, то значение верхней границы (рекорд) обновляется.

Описанный алгоритм реализован с использованием потоковой модели NPTL (Native POSIX Thread Library [5]) под CentOS 6.3. Каждый из потоков осуществляет вычисление своей α_m -ветви, исходящей из корня дерева.

На задаче приведения 103-мерной решётки ВКЗ-методом с размером блока $\beta = 52$ на четырёх потоках, выполняемых на AMD Phenom 965/8 Gb DDR2-800, он продемонстрировал трёхкратное ускорение в сравнении с последовательным ВКЗ-решателем `fp111-4.0.1` [6]. Предложенный алгоритм применялся в составе решателя, который занял 1-е и 6-е места на международных конкурсах поиска коротких векторов [7] для норм $A = m \cdot \det(L)^{1/m}$ и $A = 1,05 \frac{\Gamma(m/2 + 1)^{1/m}}{\sqrt{\pi}} \det(L)^{1/m}$ соответственно.

ЛИТЕРАТУРА

1. Schnorr C. P. Block reduced lattice bases and successive minima // *Combinatorics, Probability and Computing*. 1994. V. 3. P. 507–522.
2. Усатюк В. С. Реализация параллельных алгоритмов ортогонализации в задаче поиска кратчайшего базиса целочисленной решетки // *Прикладная дискретная математика. Приложение*. 2012. № 5. С. 120–122.
3. Kannan R. Improved algorithms for integer programming and related lattice problems // *Proc. STOC'83*. New York, NY, USA, 1983. P. 193–206.
4. Hanrot G. and Stehle D. Improved analysis of Kannan's shortest lattice vector algorithm // *LNCS*. 2007. V. 4622. P. 170–186.
5. Kerrisk M. *The Linux programming interface: a Linux and UNIX system programming handbook*. San Francisco, USA: No Starch Press, 2010. 1552 p.
6. <http://perso.ens-lyon.fr/damien.stehle/fp111/index.html> — Приложение fp111. 2013.
7. <http://www.latticechallenge.org/ideallattice-challenge/index.php> — Ideal lattice challenge (SVP, Approx-SVP). 2012.

УДК 004.627

РАСПАРАЛЛЕЛИВАНИЕ АЛГОРИТМА ДЕКОДИРОВАНИЯ СТАНДАРТА СЖАТИЯ ВИДЕОДАННЫХ H.265/HEVC

Р. И. Черняк

Рассматривается возможность параллельной реализации декодера в новом стандарте сжатия цифровых видеопоследовательностей H.265/HEVC. Предложены два способа параллелизации декодера, эффективность которых обоснована теоретически и показана экспериментально.

Ключевые слова: видеокодирование, сжатие цифрового видео, распараллеливание алгоритмов декодирования.

Новейший стандарт сжатия видео H.265/HEVC [1], предложенный в 2013 г. совместно группой экспертов по видеокодированию VCEG (Video Coding Experts Group) и экспертной группой по движущемуся изображению MPEG (Moving Picture Experts Group), представляет собой очередной шаг в технологиях видеокомпрессии. Он призван заменить широко используемый в настоящее время стандарт H.264/AVC [2]. Ключевым требованием, предъявляемым к HEVC, является двукратное уменьшение объёма сжатого видео (при фиксированном качестве) по сравнению с AVC. Для достижения данной цели разработчиками стандарта предложены новые и доработаны существующие подходы к сжатию видео. В результате цель достигнута, однако, в связи с усложнением логики работы кодека, значительно увеличилась его алгоритмическая сложность. Поскольку в приложениях критически важной является задача обеспечения кодирования и декодирования видеопотока в реальном времени (около 25 кадров в секунду), возникает задача эффективной реализации стандарта HEVC. Одним из наиболее действенных на сегодня способов увеличения скорости работы кодека на современных устройствах является параллельная реализация некоторых его алгоритмов. В данной работе речь пойдёт о параллельных алгоритмах декодирования в рамках стандарта H.265/HEVC.

1. Краткое описание кодека H.265/HEVC

Как и в предыдущих стандартах видеокомпрессии, в HEVC кодирование видеопоследовательности осуществляется по кадрам. Базовой единицей кодирования является CU (*coding unit*) — квадратный блок 16×16 , 32×32 или 64×64 пикселей. В первом приближении алгоритм кодирования можно описать следующим образом. Входящий кадр разбивается на CU, затем последовательно кодируется каждый из этих блоков. Более формально, кодек HEVC реализует классическую схему «гибридного» кодирования [3]. Не вдаваясь в её детали, отметим лишь, что завершающей стадией кодирования каждого CU является его энтропийное сжатие арифметическим кодером SBAC (*Syntax-based context-adaptive Binary Arithmetic Coding*). Соответственно первым шагом декодирования является выделение из битового потока данных, соответствующих очередному CU, их энтропийное декодирование, а затем окончательное восстановление (*реконструирование*). SBAC представляет собой контекстно-адаптивный энтропийный кодер, это значит, что при кодировании (декодировании) очередного CU используется информация о состоянии кодера в определённый момент (*контекст*). При обработке каждого CU видеопоследовательности контекст изменяется. Таким образом энтропийное кодирование и декодирование кадра осуществляется последовательно по всем CU.

2. Способы распараллеливания

В рамках данной работы предложены два механизма распараллеливания декодера HEVC. Первый представляет собой универсальный способ, который подходит для любых видеоданных, закодированных HEVC. Второй использует новую особенность стандарта H.265, в связи с этим он может быть использован только на входных видеоданных, закодированных специальным образом.

2.1. Распараллеливание по строкам

При декодировании типичного кадра Full HD-видео (кадр размера 1920×1080 пикселей) энтропийное декодирование занимает около 10 % времени работы кодека. Реконструирование изображения при этом занимает около 65 %; оставшиеся 25 % — вспомогательные операции инициализации данных. Энтропийное декодирование осуществ-

ляется сугубо последовательно; реконструирование может выполняться параллельно. Единственным требованием в этом случае является следующее.

Требование 1. Для корректной работы декодера при реконструировании CU на позиции (i, j) необходимо, чтобы уже были реконструированы четыре его «соседа»: $(i - 1, j)$; $(i - 1, j - 1)$; $(i, j - 1)$ и $(i - 1, j + 1)$.

Рис. 1 иллюстрирует данное условие. Темным цветом выделены уже реконструированные CU, светло-серым — находящиеся в процессе реконструкции, белым — ожидающие своей очереди. В данном примере реконструирование осуществляется тремя вычислителями параллельно.

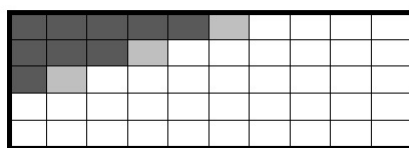


Рис. 1. Параллельное реконструирование CU в кадре

Предлагаемый механизм распараллеливания HEVC-декодера основывается на следующих двух идеях:

- 1) отделение процесса энтропийного декодирования от реконструирования;
- 2) распараллеливание реконструирования с учетом требования 1.

Предлагаемый алгоритм можно описать следующим образом. Пусть имеется один основной поток выполнения и пул из $(n - 1)$ рабочих потоков. Пусть основной поток последовательно осуществляет энтропийное декодирование кадра; как только очередная строка декодирована энтропийным декодером, пулу потоков передается очередная задача на реконструирование этой строки. В свою очередь, каждый рабочий поток из пула производит проверку условия из требования 1 и, если оно выполнено, реконструирует очередной CU. Когда энтропийное декодирование кадра завершено, основной поток дожидается завершения работы всех потоков пула. Поскольку энтропийное декодирование осуществляется значительно быстрее, чем реконструирование, достаточно быстро будут задействованы все доступные рабочие потоки.

2.2. Распараллеливание по тайлам

Одним из новшеств стандарта H.265/HEVC стало обеспечение возможности параллельного энтропийного декодирования кадра. В общем случае SBAC осуществляет последовательное декодирование, однако разработчиками стандарта предложены способы распараллеливания этой процедуры. Разбитый на CU кадр представляется в виде матрицы $n \times t$, каждым элементом которой является множество CU. Такое разбиение называется *тайловым*, а элемент матрицы — *тайлом*. В начале каждого тайла в качестве контекста для энтропийного кодека используется значение по умолчанию. Таким образом, можно осуществлять процедуры кодирования и декодирования параллельно. Рис. 2 иллюстрирует возможное тайловое разбиение кадра и способ его декодирования.

В данном примере кадр разбивается на тайлы матрицей 2×2 . Дальнейшее декодирование осуществляется параллельно в каждом тайле. Такой способ позволяет распараллелить как энтропийное декодирование, так и реконструирование кадра. Таким образом, распараллеливание по тайлам предпочтительнее, чем по строкам, однако наличие тайлов — это специфичная настройка кодека, которую нельзя гарантировать, что мешает использовать данный подход в общем случае.

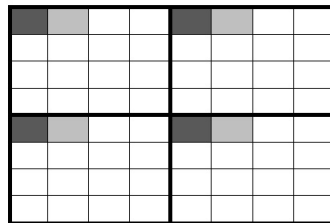


Рис. 2. Параллельное декодирование кадра по тайлам

Следует отметить, что поскольку на границе тайлов происходит сброс контекста энтропийного кодека, использование большого количества тайлов влечёт некоторое снижение качества его работы.

3. Эксперименты

В экспериментах в качестве входных данных использовались три различных видео-последовательности: анимированные фильмы Big Buck Bunny [4] и Sintel [5], свободно распространяемые организацией Blender Foundation [6], и видеозапись соревнований по серфингу, проходивших в 2012 г. во Вьетнаме [7]. Каждый видеоролик представлен в нескольких разрешениях: Big Buck Bunny — 1920×1080 (1080p), 1280×720 (720p), 640×360 (360p); Sintel — 1920×816 (816p), 1280×408 (408p), 640×272 (272p); Serfing — 1280×720 (720p), 640×360 (360p). Длительности роликов составляют 14315, 8883 и 21313 кадров соответственно.

Эксперименты проводились для трёх алгоритмов декодирования: 1) последовательного; 2) параллельного по строкам; 3) параллельного по тайлам. В качестве аппаратной платформы использовался ПК с четырёхъядерным процессором Intel Core I7 3770 и 8GB ОЗУ. В табл. 1 приведены результаты экспериментов. Скорость декодирования представлена в кадрах в секунду.

Т а б л и ц а 1

Скорость декодирования различных входных данных последовательным и параллельными алгоритмами

Входные данные	Последовательное	Парал. по строкам	Парал. по тайлам
Big Buck Bunny 1080p	20,4	38,7	43,1
Big Buck Bunny 720p	44,6	79,1	89,5
Big Buck Bunny 360p	159	201,6	242,6
Sintel 816p	23,4	47,5	52,6
Sintel 408p	83,3	128,4	154,4
Sintel 272p	172,3	217,5	263,1
Serfing 720p	23,8	50,5	56,2
Serfing 360p	80,0	123,4	155,8

Согласно результатам экспериментов, наибольшее ускорение наблюдается при применении тайловой модели распараллеливания на видеопоследовательностях высоких разрешений. С убыванием размера кадра ускорение уменьшается.

Вычислим теоретическое ускорение для тайловой и строковой моделей распараллеливания при декодировании Full HD-видео. Энтропийное декодирование, реконструирование и вспомогательные операции в этом случае занимают 10, 65 и 25 % времени декодирования соответственно. Пусть t — общее время последовательного декодирования, тогда $0,65t$ — время последовательного реконструирования. При строковом

распараллеливании на четыре ядра время реконструирования уменьшается в 4 раза. Таким образом, общее время декодирования может быть вычислено по формуле $0,65t/4 + 0,1t + 0,25t = 0,5125t$. Поскольку скорость декодирования обратно пропорциональна времени, максимальный коэффициент ускорения декодирования составляет 1,95. Вычисленный аналогичным образом коэффициент ускорения для тайлового распараллеливания составляет 2,29. Заметим, что экспериментально полученные значения ускорений несколько меньше теоретических. Это объясняется тем, что при обеих моделях распараллеливания данные зависят друг от друга. В случае строковой модели эта зависимость определяется требованием 1, в случае тайловой — потенциально различной сложностью каждого тайла.

Как отмечалось ранее, с ростом количества тайлов ухудшается эффективность работы энтропийного кодера. Проведён эксперимент с целью определения степени ухудшения. Одно и то же видео кодировалось в несколько тайлов при прочих одинаковых настройках кодирования. Табл. 2 иллюстрирует изменение качества видео, выраженного в виде пикового отношения сигнала к шуму (*PSNR*), и эффективности его сжатия (*bitrate*) в зависимости от количества тайлов. Видно, что ухудшение качества видео и степени сжатия пренебрежимо мало даже при кодировании каждого кадра в восемь тайлов. Эксперимент проводился для видео с разрешением 720p.

Таблица 2

Зависимость качества видео и степени сжатия от количества тайлов

Количество тайлов	1	2	3	4	5	6	7	8
Bitrate (kbps)	1456,2	1466,2	1474,4	1486,2	1494,9	1501,7	1507,7	1519,6
PSNR (dB)	41,448	41,444	41,442	41,440	41,437	41,434	41,433	41,432

Таким образом, теоретически и экспериментально показано, что использование модели тайлового распараллеливания даёт выигрыш по сравнению со строковым. Вместе с тем тайловая модель не может быть использована для произвольных данных. Представляется целесообразным использовать тайловое распараллеливание в случае, если входное видео закодировано в несколько тайлов, в противном случае стоит использовать строковый алгоритм. В случае тайлового распараллеливания строковый метод может быть использован внутри каждого тайла. Такой подход представляет интерес при декодировании на многоядерных устройствах с целью обеспечения высокой скорости декодирования путём максимальной загрузки каждого ядра.

ЛИТЕРАТУРА

1. Bross B., Han W.-J., Ohm J.-R., et al. High efficiency video coding (HEVC) text specification draft 10 (for FDIS & Last Call). 12th Meeting: Geneva, CH, 14–23 Jan. 2013
2. ISO/IEC 14496-10:2003. Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding.
3. Sullivan G. J., Ohm J.-R., Han W.-J., and Wiegand T. Overview of the High Efficiency Video Coding (HEVC) standard // IEEE Trans. Circuits and Systems for Video Technology. 2012. V. 22. No. 12. P. 1649–1668.
4. <http://www.bigbuckbunny.org/> — Big Buck Bunny. 2013.
5. <http://www.sintel.org/> — Sintel, the Durian Open Movie Project. 2013.
6. <http://www.vungtau-city.com/?p=1558> — Russia Surfing Cup and International Kitesurfing Competitions in Vung Tau | Vung Tau City Portal. 2013.
7. <http://www.blender.org/> — blender.org — Home of the Blender project — Free and Open 3D creation software. 2013.