doi:10.3969/j.issn.1001-893x.2014.10.022
引用格式:邸金红,张克新,祁跻,等.基于 Syntax 级分组和多线程处理的 HEVC 熵编码并行算法[J].电讯技术,2014,54(10):1435-1440.[DI Jin-hong,ZHANG Ke-xin,Qi Ji,et al. A Parallel Algorithm for HEVC Entropy Coding Based on Syntax-level Grouping and Multi-thread Processing[J]. Telecommunication Engineering,2014,54(10):1435-1440.]

基于 Syntax 级分组和多线程处理的 HEVC 熵编码并行算法*

邸金红[1,**],张克新[1],祁　跻[2],张鑫明[3]

(1.郑州航空工业管理学院 电子通信工程系,郑州 450015;2.阿拉巴马大学,美国 塔斯卡卢萨 870118;
3.中国舰船研究院, 北京 100192)

摘　要:新一代视频编码标准 HEVC 获得了较高的编码效率,但是同时需要较大的计算量。HEVC 并行算法能够提高编码速度,如何开发适用于多核处理器的并行编码算法对于满足高清视频实时传输和大规模共享具有十分重要的意义。提出了一种基于 Syntax 级分组和多线程处理的 HEVC 熵编码并行算法。该算法首先将 HEVC 中一个编码树单元的编码信息按照语法元素进行分组;其次,根据编码块数据间的相关性构建 Syntax 级并行编码器;然后结合多线程技术实现 HEVC 帧级编码的并行计算。实验结果表明,在编码图像的主客观质量上没有太大损失的情况下,该并行算法框架与传统的串行算法框架相比具有 65% ~70% 的加速效果。

关键词:高效视频编码;并行算法;Syntax 级;多线程
中图分类号:TN919.8　　**文献标志码**:A　　**文章编号**:1001-893X(2014)10-1435-06

A Parallel Algorithm for HEVC Entropy Coding Based on Syntax-level Grouping and Multi-thread Processing

DI Jin-hong[1],ZHANG Ke-xin[1],QI Ji[2],ZHANG Xin-ming[3]

(1. Department of Electronic and Communication Engineering,Zhengzhou Institute of Aeronautical Industry Management,Zhengzhou 450015,China;2. The University of Alabama,Tuscaloosa 870118,USA;
3. China Ship Research and Development Academy,Beijing 100192,China)

Abstract:As a new generation video coding standard,High Efficiency Video Coding(HEVC) achieves higher compression efficiency,but also requires a large amount of calculation. The parallel HEVC encoding algorithm is important to improve the encoding speed,therefore the design of parallel coding algorithm suitable for multi-core processors is of great significance to meet the requirements of High Density(HD) video real-time transmission and large-scale sharing. In this paper,a parallel entropy-coding framework based on Syntax-level grouping and multi-thread processing is presented. Firstly,the coding messages of a Coding Tree Unit(CTU) in HEVC are grouped according to the syntax elements;secondly,the parallel encoder based on Syntax-level is built according to the correlation between the data blocks;finally,frame-level parallel coding in HEVC is achieved by combining with the multi-threaded calculation. Experimental results show that the encoding speed of the proposed solution is 65% ~70% faster than that of the traditional serial architecture,meanwhile the video quality doesn't lose much.

Key words:high efficiency video coding;parallel algorithm;syntax-level;multi-thread

*　**收稿日期**:2014-06-16;**修回日期**:2014-08-29　　Received date:2014-06-16;Revised date:2014-08-29
基金项目:国家自然科学基金资助项目(61271190);河南省科技厅科技攻关项目(142102210506)
Foundation Item:The National Natural Science Foundation of China(No. 61271190);The Scientific and Technological Research Projects in Henan Province(142102210506)
**　**通讯作者**:jinhongdi@163.com　　**Corresponding author**:jinhongdi@163.com

1 引 言

高效视频编码(High Efficiency Video Coding, HEVC)是由 ISO 的动态图像编码专家组(Moving Pictures Experts Group/Motion Pictures Experts Group, MPEG)和 ITU 的视频编码专家组(Video Coding Experts Group, VCEG)共同提出和制定的下一代视频编码标准。作为现行视频编码标准 H. 264/AVC的继任者, HEVC 的目标是较 H. 264/AVC 有 2 倍的压缩效率并支持到 7680×4320 的视频分辨率[1]。HEVC 以算法复杂度的增加来换取压缩效率的进一步提高[2-3],然而,算法复杂度越高,运算消耗的资源也就越大,从目前 HEVC 标准测试代码 HM 的运算时间来看,难以满足实际应用中实时编码的需求。针对这一难题,面向并行多核/众核处理器的视频编码并行算法为大数据量运算问题提供了一种重要的解决手段,也是当前多媒体技术领域研究的热点之一。

不同于 H. 264/AVC 中分别应用与基本档和高档编码配置的两种熵编码算法, HEVC 仅采用一种熵编码模式,即基于上下文的自适应二进制算术编码(Context - Adaptive Binary Arithmetic Coding, CABAC)[4]。熵编码模块以其与其他模块的紧耦合性和数据之间的强相关性,成为并行化处理的瓶颈。针对 HEVC 测试模型, K. Misra[5]等提出了 Entropy Slice 的概念,它具有较高的并行粒度,并且良好地解决了负载不均衡的问题。Clare[6]提出行波并行处理算法(Wave-front Parallel Processing, WPP),波面用于实现帧层次的并行,相当于多帧多线程处理。随后在 WPP 的基础上提出相应的 CABAC 编码方法[7]。Arild[8]提出一种对宏块分割的新结构 Tiles,有利于减小并行中宏块分割所带来的失真。对于 CABAC,随着下文模型动态更新机制的引入,使得整个熵编码过程成为了一个高度紧耦合的串行过程。强行进行图像块的划分,必然会导致编码性能的下降。因此,基于现有串行的编码框架体系,使用并行计算加速编码的总体效果并不理想。

为了提高 HEVC 编码速度,本文依据并行计算的适用特点,提出了一种基于 Syntax 分组流水线处理和多线程处理的并行算法,并在 HM 10.0 平台上进行了仿真。

2 HEVC 熵编码并行策略分析

HEVC 熵编码的并行策略主要有 Bin 级并行处理、Slice 级并行处理和 Syntax 级并行处理 3 种。

2.1 Bin 级并行处理

Bin 级的并行策略主要是在二进制算术编码部分进行并行处理。输入的句法元素经过二值化被映射成一个二进制序列,这个二进制序列在经过上下文建模和概率估计后,以比特为单位被分拆成若干部分分别进行算术编码,编码完后再进行组合最终形成输出码流。Bin 级并行处理策略的并行粒度取决于上下文建模和概率估计的处理能力,因此对提高系统性能方面影响不大,有时反而会引起系统性能的下降。

2.2 Slice 级并行处理

根据 H. 264/AVC 中 Slice 分割的思想, HEVC 在保留传统 Slice 的基础上引入了 Entropy Slice[5,9]的并行编码策略。与传统的 Slice 不同的是, Entropy Slice 将邻近编码片的位置信息写入码流,使其在重建图像的时候可以利用邻近 Slice 中包括运动估计及帧内预测在内的编码信息,编码性能得到进一步提高。Entropy Slice 允许在一个 Slice 内部再切分成多个 Entropy Slices,这样熵编码器可以并行编码,从而提高了并行处理能力,使编码器的负载更加均衡。利用 Entropy Slice 并行方案,可以同时调动 CPU 和 GPU 进行运算,使整个编码系统的运算能力得以充分发挥。无论是传统 Slice 还是 Entropy Slice,其并行粒度依然局限于编码片的划分。对于当今大规模并行计算的需求,依然有很大的差距。随着 HEVC 编码框架的完善和新的方法不断涌现,该技术已不再被标准所推崇[6-8]。

2.3 Syntax 级并行处理

Syntax 级的并行策略不将一组二进制符号分配到不同的熵编码器中,也不按照实际编码单元进行分割,而是按照语法元素的不同进行分割从而实现并行处理。在现行的熵编码框架中,对于每种语法元素,都使用特定的上下文模型进行建模,因此,按照语法元素进行分类,可以使这些上下模型进行尽可能多的训练过程,从而使编码概率估计更加精确,因此本文采用了 Syntax 级并行策略。此外,无论是大型数据工作站还是个人计算机,其核心 CPU 都具备多线程架构,并且随着 CPU 技术的发展,其核心数会越来越多。多线程技术是并行计算的一个重要组成部分,因此可以利用 CPU 的多线程计算能力来实现 HEVC 帧级的并行编码。结合两者的优点,本

文提出了基于 Syntax 级分组和多线程处理的并行算法,下面详细介绍具体实现过程。

3　提出的熵编码并行算法

3.1　Syntax 级分组

与 H.264/AVC 相类似,HEVC 中的语法元素同样可以进行分组以实现 Syntax 级的并行处理。对于 HEVC 中一个编码树单元(Coding Tree Unit, CTU)里的编码信息,按照语法元素进行划分可以分为以下几组:编码块划分信息 SpiltFlag,编码方式 PredMode,块划分方式 PartSize,预测信息 PredInfo,变换系数块标志位 Cbf,非零变换系数位置信息 SigMap 和非零变换系数信息 Coeff。具体来说,PredInfo 包括 IntraDirLumaAng、IntraDirChroma 等帧内角度预测信息和 MergeFlag、SkipFlag、MergeIndex、RefFrmIdx 和 Mvd 等帧间运动估计信息;SigMap 包括 LastSignificantX/Y、SignificantCoeffFlag、SignificantCoeffGroupFlag 等非零变换系数位置信息;Coeff 包括 C1Flag、C2Flag、AbsCoeff 和 CoeffSigns 等非零变换系数幅值和符号信息。

这些语法元素之间具有信息间的依赖性,如图 1 所示。需要注意的是语法元素 SkipFlag,该语法元素和其对应的语法元素 MergeIndex 虽然也属于 PredInfo 一组,但并不同其他属于 PredInfo 的语法元素一样依赖于 PartSize 和 PredMode 的信息,因此,需要将此语法元素单独分离出来。基于上述分析,构建一个如图 2 所示的 Syntax 级并行编码器。

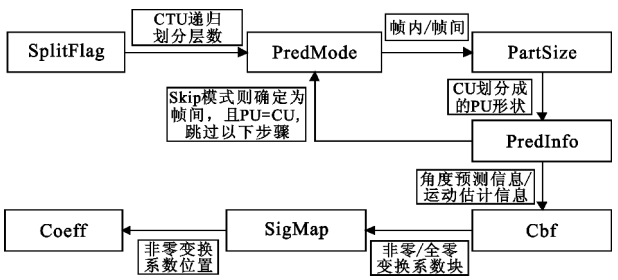

图 1　HEVC 编码块中数据间的相互关系

Fig. 1 The relationship between data blocks in HEVC

将 CTU 中的语法元素进行分组后,需要在每组语法元素之前设置一个标志位(StartFlag)以与其他组进行区分。在本文提出的并行算法中,利用一元码来设置 StartFlag。对于图 2 中 9 组语法元素,按照流水线的先后处理顺序分别赋予 0、10、110、1110、11110、111110、1111110、11111110 和 111111110 作为标志位码字。

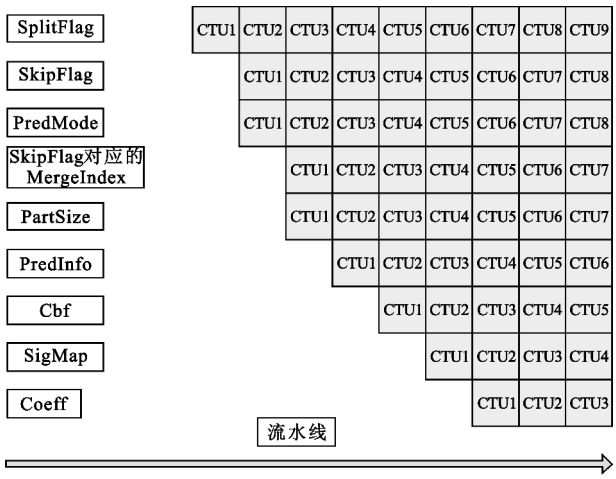

图 2　HEVC 中 Syntax 级并行编码器结构

Fig. 2 The parallel structure based on syntax-level in HEVC

3.2　多线程处理

在 HEVC 中,如果采用全 I 帧编码,则各帧之间编码数据相互独立,各帧在编码时不需要数据间的相互通信。在本文提出的多线程编码方法中,需要单独开辟一个线程进行码流的排序和拼接。以八线程处理为例,首先将视频序列的前 7 帧依次送入第 1 ~7 个线程后分别进行编码,编码完毕后这 7 个线程进行同步,并输出码流到第 8 个线程,在第 8 个线程中进行码流的排序和拼接。同时,将视频序列的第 8 ~14 帧再依次送入到前 7 个线程中进行编码,以后依次类推,整个编码过程如图 3 所示。

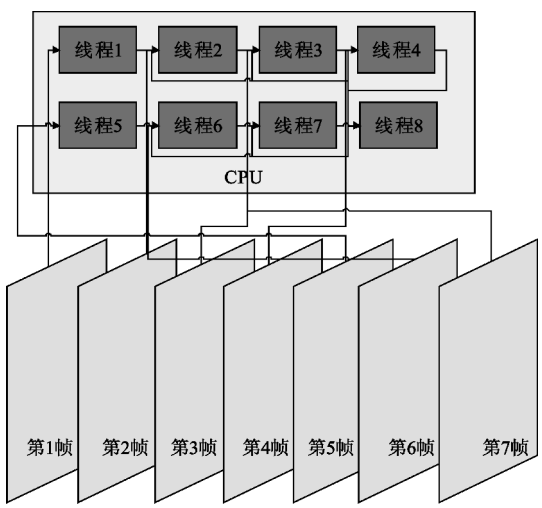

图 3　全 I 帧八线程编码方案

Fig. 3 Eight-thread coding scheme for I frame

如果采用 I/B/P 帧编码,则各帧之间数据不相互独立,B/P 帧进行编码时都需要利用参考帧的信息,因此需要数据间的相互通信。在本文提出的多线程编码方法中,不仅需要单独开辟一个线程进行码流的排序和拼接,还需开辟一个线程进行参考帧管理。以八线程处理为例,首先将视频序列的前 6 帧依次送入第 1 ~6 个线程,然后在第一个线程编码第一帧,进而将编码完后的数据信息送入用于参考帧管理的第 7 个线程,将码流送入用于输出码流的第 8 个线程,接着开启第 2 ~6 帧的编码。在这 5 帧编码期间,第 7 个线程需要不断地进行数据通信和协调同步处理,以完成帧间编码运动估计,然后这 5 帧将码流输入到第 8 个线程,在第 8 个线程中进行码流的排序和拼接。同时,将视频序列的第 7 ~ 12 帧再依次送入到前 6 个线程中进行编码,以后依次类推,整个编码过程如图 4 所示。

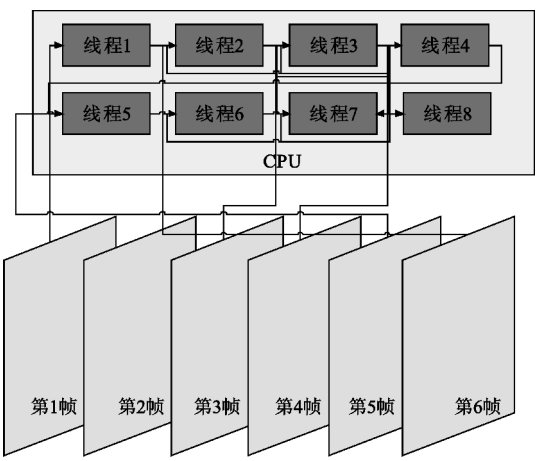

图 4 I/B/P 帧八线程编码方案

Fig. 4 Eight-thread coding scheme for I/B/P frame

HEVC 中参考帧列表管理非常复杂,如果将完整的双向预测放入一个线程中进行管理,就会导致该线程负荷过重,反而会降低处理速度。因此,为了提高多线程计算效率,本文提出的方法中帧间预测中仅采用前向预测。

4 实验结果及分析

算法测试的硬件平台所用的 CPU 型号为 Intel Core i7,主频为3.5 GHz,核心数为四核,可支持八线程计算。由于是并行算法,实验主要是从时间加速方面来验证算法的有效性,同时也考虑了视频压缩的主客观质量,其中,客观质量是用 PSNR 来衡量的。在验证并行算法效果时,对于全 I 帧并行处理,选用官方编码器中只支持全 I 帧编码的 All Intra (AI)设置与其进行比较;而对于 I/B/P 帧并行处理,由于算法中帧间预测仅采用前向预测,因此,选用官方编码器中只支持前向预测的 LowDelay(LD)设置与其进行比较。

仿真实验选用 HEVC 官方标准视频序列库 Class B (1920 × 1080, 1080P)中的 Kimono、ParkScene、BQTerrace、BasketballDrive 序列和 Class C (832 × 480, WVGA)中的 BasketballDrill、BQMall、RaceHorses 和 PartyScene 序列,这两组序列分别在 AI、LD 两种编码条件下进行测试。表 1 和表 2 为 4 个 1080P 序列分别在 AI、LD 设置下的性能比较结果,表 3 和表 4 为 4 个 WVGA 序列分别在两种不同设置下的性能结果比较。

表 1 AI 设置编码 1080P 序列串行计算与并行计算速度性能比较

Table 1 The performance comparison between serial computing and parallel computing for 1080P(AI)

测试序列(1080P)	HM10.0 编码平均时间/(s/frame)	并行算法编码平均时间/(s/frame)	编码时间减少/(%)	BD-rate/(%)
BasketballDrive(500 帧)	97.77	28.83	70.5	0.07
BQTerrace(600 帧)	109.12	38.29	64.9	0.31
Kimono(240 帧)	92.38	27.71	70.4	0.03
ParkScene(240 帧)	92.84	33.32	64.1	0.74

表 2 LD 设置编码 1080P 序列串行计算与并行计算速度性能比较

Table 2 The performance comparison between serial computing and parallel computing for 1080P(LD)

测试序列(1080P)	HM10.0 编码平均时间/(s/frame)	并行算法编码平均时间/(s/frame)	编码时间减少/(%)	BD-rate/(%)
BasketballDrive(500 帧)	380.55	99.42	73.9	1.64
BQTerrace(600 帧)	218.67	96.58	55.8	1.46
Kimono(240 帧)	260.13	96.26	62.9	1.81
ParkScene(240 帧)	202.12	51.84	74.3	2.68

表 3　AI 设置编码 WVGA 序列串行计算与并行计算速度性能比较

Table 3 The performance comparison between serial computing and parallel computing for WVGA(AI)

测试序列(WVGA)	HM10.0 编码平均时间/(s/frame)	并行算法编码平均时间/(s/frame)	编码时间减少/(%)	BD-rate/(%)
BasketballDrill(500 帧)	43.44	8.80	79.7	0.48
BQMall(600 帧)	29.85	9.29	68.9	0.38
RaceHorses(300 帧)	48.72	11.66	76.1	0.64
PartyScene(500 帧)	37.72	9.29	75.2	0.30

表 4　LD 设置编码 WVGA 序列串行计算与并行计算速度性能比较

Table 4 Theperformance comparison between serial computing and parallel computing for WVGA(LD)

测试序列(WVGA)	HM10.0 编码平均时间/(s/frame)	并行算法编码平均时间/(s/frame)	编码时间减少/(%)	BD-rate/(%)
BasketballDrill(500 帧)	42.92	15.48	63.9	1.18
BQMall(600 帧)	55.69	22.76	59.1	2.14
RaceHorses(300 帧)	106.32	28.84	72.9	1.31
PartyScene(500 帧)	103.69	33.68	67.5	1.85

由表 1 ~4 可见,从编码时间上来看,本文提出的并行算法与 HM10.0 相比,在 AI 设置上平均有 70% 左右的编码时间节省,在 LD 设置上平均有 65% 左右的编码时间节省。从这一数据可以看出,多线程计算并不能达到完全的成倍加速,其原因在于线程间数据需要相互通信,线程间并行处理需要不断进行同步,这些过程都需要消耗一定的时间。从编码图像的 BD-rate 来看,本章提出的并行算法与 HM10.0 相比,在 AI 设置上平均有 0.4% 左右的 BD-rate 损失,在 LD 设置上平均有 1.8% 左右的 BD-rate 损失,其原因在于并行处理还是损害了编码数据之间的相关性,从而使得熵编码的上下文建模过程和概率估计过程精确度下降。而由图 5 可见,应用本节的并行算法后,输出的编码图像相比于 HM10.0 在主观质量上并没有太大的变化。

(a)HM10.0编码结果

(b)本文并行算法编码结果

图 5　Kimono 序列

Fig. 5 Coding results of Kimono sequence

5　结束语

为了提高 HEVC 编码系统的运算速度,本文提出一种基于 Syntax 分组流水线处理和多线程计算的并行算法。该算法借鉴 H.264/AVC 中 Syntax 级的并行算法思路,将码流中的语法元素分组并以流水线形式实现并发处理,同时,采用多线程并行处理技术,充分利用 CPU 的核心数量,合理进行资源调度。将所提出的并行算法与传统的串行算法进行比较,验证了该算法的有效性。但是当前并行算法框架加速比不够高,远远不能满足实时视频编码的要求,需要进一步研究视频编码中的各种编码数据之间的依赖关系,找到更优的压缩效率与计算速度的折衷方案,加大整个系统的并行计算粒度。

参考文献:

[1] JCTVC-A124,Samsung's response to the call for proposals on video compression technology[S].

[2] Correa G,Assuncao P, Agostini L,et al. Performance and Computational Complexity Assessment of High-Efficiency Video Encoders[J]. IEEE Transactions on Circuits and Systems for Video Technology,2012,22(12):1899-1909.

[3] Frank B,Bross B,Sühring K,et al. HEVC Complexity and Implementation Analysis[J]. IEEE Transactions on Circuits and Systems for Video Technology,2012,22(12):1685-1696.

[4] Vivienne S, Budagavi M. High Throughput CABAC Entropy Coding in HEVC[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2012, 22(12): 1778-1791.
[5] JCTVC-B111, Entropy slices for parallel entropy coding[S].
[6] JCTVC-F274, Wavefront Parallel Processing for HEVC Encoding and Decoding[S].
[7] JCTVC-F275, Wavefront and CABAC Flush: Different Degrees of Parallelism Without Transcoding[S].
[8] JCTVC-F335, Tiles[S].
[9] JCTVC-D070, Lightweight slicing for entropy coding[S].

作者简介:

邸金红(1980—),女,河南商水人,2012 年于北京邮电大学获博士学位,现为郑州航空工业管理学院讲师,主要研究方向为多媒体通信、视频编码与传输;

DI Jin-hong was born in Shangshui, Henan Province, in 1980. She received the Ph. D. degree from Beijing University of Posts and Telecommunications in 2012. She is now a lecturer. Her research interests include multimedia communications, video coding and transmission.

Email: jinhongdi@163.com

张克新(1986—),男,河南登封人,分别于 2007 年和 2012 年获华南理工大学工学学士学位和工学博士学位,现为郑州航空工业管理学院讲师,主要研究方向为视频编码、多媒体技术与应用;

ZHANG Ke-xin was born in Dengfeng, Henan Province, in 1986. He received the B. S. degree and the Ph. D. degree from South China University of Technology in 2007 and 2012, respectively. He is now a lecturer. His research interests include video coding, multimedia technology and its application.

祁　跻(1989—),男,2014 年于北京邮电大学获硕士学位,现为美国阿拉巴马大学博士研究生,主要研究方向为视频编码;

QI Ji was born in 1989. He received the M. S. degree from Beijing University of Posts and Telecommunications in 2014. He is currently working toward the Ph. D. degree. His research direction is video coding.

张鑫明(1986—),男, 2012 年于北京邮电大学获博士学位,现为中国舰船研究院工程师,主要研究方向为认知无线电。

ZHANG Xin-ming was born in 1986. He received the Ph. D. degree from Beijing University of Posts and Telecommunications in 2012. He is now an engineer. His research direction is cognitive radio.