Research and implementation of HEVC distributed parallel

Yuwen Chen[1,a], Ju Zhang[1,b], Jin Zhang[2,c], Kunhua Zhong[2,d], XueLiang Zhao[1,e], Qilong Sun[1,f*]

[1]High Performance Computing Center CIGIT CAS, Chongqing, China

[2]Network Center CIGIT CAS,Chongqing, China

[a]chenyuwen@cigit.ac.cn, [b]zhangju@cigit.ac.cn, [c]zhangjin@cigit.ac.cn, [d]zkh@cigit.ac.cn, [e]zhaoxueliang@cigit.ac.cn

[f]sunqilong@cigit.ac.cn

Keywords: HEVC; MPI; super-computing

Abstract. High Efficiency Video Coding (HEVC) standard achieves double compression efficiency compared to H.264/AVC with the adoption of more flexible coding structure and advanced coding tools. On the other hand, the coding mode space is too large and it's very time consuming. With the development of many-core computing architecture, parallelizing HEVC encoding on such platforms is an efficient approach to fulfill the high computational requirement. In this paper, we propose a highly parallel MPI method which is based on the reference software for the HEVC. Firstly we split the video into multiple computing cores, and then the video frame is compressed simultaneously. Experimental results show that our proposed parallel MPI method gets an overall speed up of more than 10x with 336 computing cores, compared with the non-parallel baseline.

Introduction

The High Efficiency Video Coding (HEVC) standard is the most recent joint video project of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) standardization organizations, working together in a partnership known as the Joint Collaborative Team on Video Coding (JCT-VC) [14]. The major video coding standard directly preceding the HEVC project was H.264/MPEG-4 AVC, which was initially developed in the period between 1999 and 2003, and then was extended in several important ways from 2003–2009. H.264/MPEG-4 AVC has been an enabling technology for digital video in almost every area that was not previously covered by H.262/MPEG-2 Video and has substantially displaced the older standard within its existing application domains. It is widely used for many applications, including broadcast of high definition (HD) TV signals over satellite, cable, and terrestrial transmission systems, video content acquisition and editing systems, camcorders, security applications, Internet and mobile network video, Blu-ray Discs, and real-time conversational applications such as video chat, video conferencing, and telepresence systems.

However, an increasing diversity of services, the growing popularity of HD video, and the emergence of beyond HD formats (e.g., 4k×2k or 8k×4k resolution) are creating even stronger needs for coding efficiency superior to H.264/MPEG-4 AVC's capabilities. The need is even stronger when higher resolution is accompanied by stereo.. An increased desire for higher quality and resolutions is also arising in mobile applications.

HEVC is developed [1,2] with the target of outperforming all existing standards, specifically doubling the compression efficiency than H.264/AVC. Although still conforming to the hybrid video coding framework, HEVC adopts enhancements in each stage. One of the most important innovations is the adoption of flexible coding structure. As shown in Fig. 1(a), a Coding Tree Unit (CTU) is split recursively into four equally sized units in a quad-tree manner and a leaf node of the resulting recursive quad-tree (Coding Tree) is called a Coding Unit (CU). For a certain CU, it is also recursively split in a quad-tree manner with leaf nodes of this tree (Transform Tree) called Transform Units (TU). Furthermore, there are many prediction unit (PU) partition modes available for a CU to flexibly perform video content prediction, as shown in Fig. 1(b). Due to the flexible block splitting in HEVC, the coding mode search space is very large for an encoder to seek for the best mode combination. Many research works have been done to accelerate HEVC. Proposals [3-5] propose efficient fast algorithms including early detection and mode skipping and all of these proposals have been incorporated in the HEVC reference software HM [6]. These methods reduce HEVC encoding complexity to some extent, but the speed up they provide is not high enough.

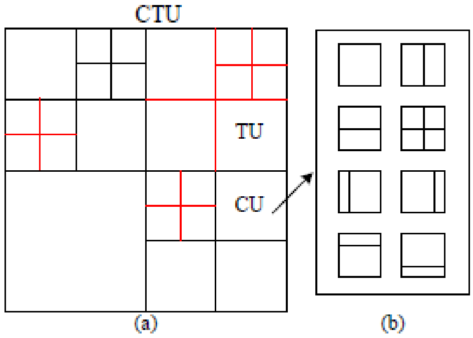


Figure1. Flexible block partitioning in HEVC. (a) CU and TU, (b) PU partition.

With the increasing development of many-core computing architectures, parallelizing HEVC encoding on such platforms is an efficient solution to fulfill the high computational requirement of HEVC encoder [7-9]. Compared with its predecessors, HEVC standard itself adopts several features that facilitate parallel codec such as wavefront parallel processing (WPP) [10], tile [11] and motion estimation region (MER) [12]. WPP has low impact on coding quality but with relatively low parallelism. [13] proposes a method namely overlapped wavefront (OWF) based on WPP which can improve the parallelism. Partitioning a picture into many tiles can provide relatively a high parallelism but this will severely drop compression efficiency. While WPP and tile facilitate parallel processing among large regions, MER defines a local region to allow parallel ME within this region, but MER only applies to ME and other parts of MD still has to computed sequentially which results in low overall throughput.

This paper is organized as follows. Section II we briefly cover the background of the reference software for the HEVC (HM). In section III our proposed parallelization strategies based on HM are introduced and implemented . In section V the experimental results are shown. At last, this paper is concluded.

Reference software for the HEVC

Reference software is being made available to provide a reference implementation of the draft HEVC standard being developed by the Joint Collaborative Team on Video Coding (JCT-VC) regrouping experts from ITU-T SG 16 and ISO/IEC SC29 WG11. One of the main goals of the reference software is to provide a basis upon which to conduct experiments in order to determine which coding tools provide desired coding performance. our proposed parallelization strategies are based on the Reference software, which will be introduce in section III

IMPLEMENTATION

Based on HM [6], the reference software of HEVC, we do not change the internal compression algorithm processes, and compress the input YUV420P format video data by segmenting it in chronological order with MPI implementation. The original YUV420P video data will be read in the form of shared memory. Each computing core reads one video data segment on a continuous period of time, and then compresses this data segment, to get the corresponding binary code stream and stored it as file. After the completion of all the cores, all the binary code streams will be combined into a single file. As shown in the diagram Figure

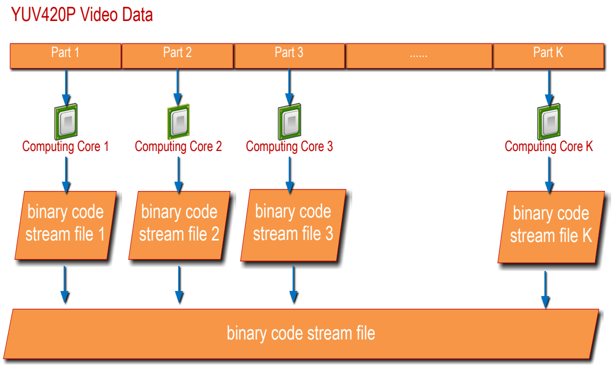
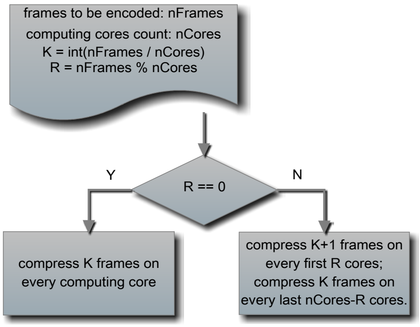


Figure2.hevc parallel processing diagram

Before compressed video ,we split the video into video frames ,and then the video frames are assigned to different computing cores and calculated according to the video frame distribution strategy design which is shown as fig.3. We assumed video frames to be encoded is nFrames; computing cores count is nCores; k is the quotient of the nFrames divided nCores; R is the remainder. If R is equal to 0, Each computing core compressed K frames. else the first R computing core compress K+1 frames,the remaining computing core compress K frames.


frames to be encoded: nFrames
computing cores count: nCores
K = int(nFrames / nCores)
R = nFrames % nCores
R == 0
Y
N
compress K frames on every computing core
compress K+1 frames on every first R cores; compress K frames on every last nCores-R cores.

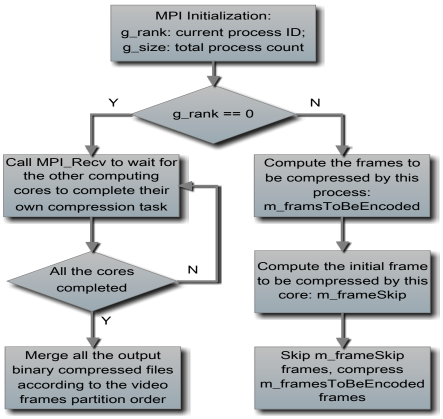

MPI Initialization:
g_rank: current process ID;
g_size: total process count
g_rank == 0
Y
N
Call MPI_Recv to wait for the other computing cores to complete their own compression task
All the cores completed
N
Y
Merge all the output binary compressed files according to the video frames partition order
Compute the frames to be compressed by this process: m_framsToBeEncoded
Compute the initial frame to be compressed by this core: m_frameSkip
Skip m_frameSkip frames, compress m_framesToBeEncoded frames

Figure 3.video frame distribution strategy Figure4.compress flow chart

According to the video frame distribution strategy and HEVC reference software HM ,we implement the distributed compression system based on super-computing cluster .The flow chart is shown as follows fig 4. Firstly supercomputing cluster Initialize the MPI to get the current process and the total number of the process. the main process waits for the other to complete the compression process. The other process compress the allocated video frames and write it into small files after the completion of video frames. The main process combined with the small files into compressed video files waiting for other processes after the completion .

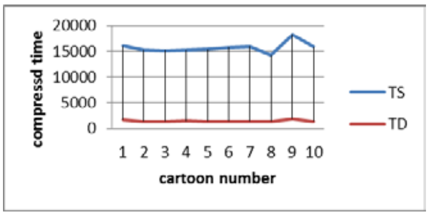
EXPERIMENTAL RESULTS

To evaluate the efficiency of our proposed parallel method, ten episode cartoon sequences which provided by XiangHong Film & TV are encoded using the HM encoder as baseline and our parallelized version. The speed up is derived as (1) in which TS and TD mean the encoding time consumed by the baseline and our parallelized encoder. Based on super-computing resources in Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, The experimental results are shown in Table I, in which the speed up is obtained by using 336 cores

$$\text{speed up}=\frac{TS}{TD} \qquad (1)$$

TABLE I. EXPERIMENTAL RESULTS OF OUR PROPOSED PARALLEL MPI METHOD.

Sequences	TS(s)	TD(s)	speed up
Survival Adventure 1	16013	1775	9
Survival Adventure 2	15165	1389	11
Survival Adventure 3	15105	1398	11
Survival Adventure 4	15165	1489	10
Survival Adventure 5	15317	1413	11
Survival Adventure 6	15771	1411	11
Survival Adventure 7	15922	1422	11
Survival Adventure 8	14167	1425	10
Survival Adventure 9	18283	1884	10
Survival Adventure 10	15862	1440	11



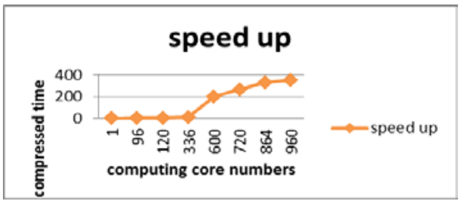


Figure 5.experimental results of our proposed parallel mpi method.

Figure 6.speed up using different number of cores

Figure 6 shows the speed up of one of sequences using different number of cores (96, 120, 336, 600, 720,864,960) . It can be seen that our proposed parallel MPI method behaves good .the more cores ,the better speed up .but when less than 960 cores are used but the curve gradually becomes flat from this point. Because With the increase of the computing cores, each computing core processing video frames is reduced .when reaches the preset minimum video frame, time-consuming tends unchanged.

Summary

In this paper, based on the super-computing resource, we propose a highly parallel MPI method which is based on the reference software for the HEVC project and implement a highly parallel HEVC version, which gets an overall speed up of more than 14x with 72 computing cores, compared with the non-parallel baseline. Experimental results show method is fast and effective.

However, this system is only a back-end system running on the server. We can provide external services with the B/S architecture in the future.

Acknowledgment

This work was financially supported by the National Science and Technology Ministry (2013BAH25F02).

References

[1] B. Bross, W.-J. Han, J.-R. Ohm, G. J. Sullivan, Y.-K. Wang, and T. Wiegand, “High Efficiency Video Coding (HEVC) text specification draft 10 (for FDIS & Last Call)”, JCTVC-L1003, Geneva, CH, Jan. 2013

[2] G. J. Sullivan, J.-R. Ohm, W.-J. Han and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) Standard”, IEEE TCSVT, 22(12), pp 1649-1668, Dec. 2012

[3] K. Choi, S.-H. Park and E. S. Jang, “Coding tree pruning based CU early termination”, JCTVC-F092, Torino, IT, July, 2011

[4] J. Yang, J. Kim, K. Won, H. Lee and B. Jeon, “Early SKIP Detection for HEVC”, JCTVC-G543, Geneva, CH, November 2011

[5] R. H. Gweon, Y.-L. Lee and J. Lim, “Early Termination of CU Encoding to Reduce HEVC Complexity”, JCTVC-F045, Torino, IT, July 2011

[6] HEVC Test Model, HM 10.1. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/branches/HM- 10.1-dev/

[7] K. Choi and Euee S. Jang, "Leveraging Parallel Computing in Modern Video Coding Standards", IEEE Multimedia, 19(3), pp 7-11, 2012

[8] Yongdong Zhang, Chenggang Yan, Feng Dai and Yike Ma, "Efficient parallel framework for H.264/AVC deblocking filter on many-core platform", IEEE TMM, 14(1), pp 510-524, 2012

[9] Chenggang Yan, Yongdong Zhang, Feng Dai, and Liang Li, "Highly Parallel Framework for HEVC Motion Estimation on Many-core Platform," DCC 2013, UT, USA, Mar. 2013

[10] G. Clare, F. Henry and S. Pateux, "Wavefront Parallel Processing for HEVC Encoding and Decoding", JCTVC-F274, Torino, IT, July 2011

[11] A. Fuldseth, M. Horowitz, Sh. Xu, A. Segall and M. Zhou, "Tiles", JCTVC-F335, JCTVC, Torino, IT, July 2011

[12] Minhua Zhou, "AHG10: Configurable and CU-group level parallel merge/skip", JCTVC-H0082, JCTVC, San Jose, CA, USA, Feb. 2012

[13] Chi Ching Chi, Mauricio alvarez-Mesa, B. Juurlink et al, "Parallel Scalability and Efficiency of HEVC Parallelization Approaches", IEEE TCSVT, 22(12), pp 1827-1838, Dec. 2012

[14] B. Bross, W.-J. Han, G. J. Sullivan, J.-R. Ohm, and T. Wiegand, High Efficiency Video Coding (HEVC) Text Specification Draft 9, document JCTVC-K1003, ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC), Oct. 2012.