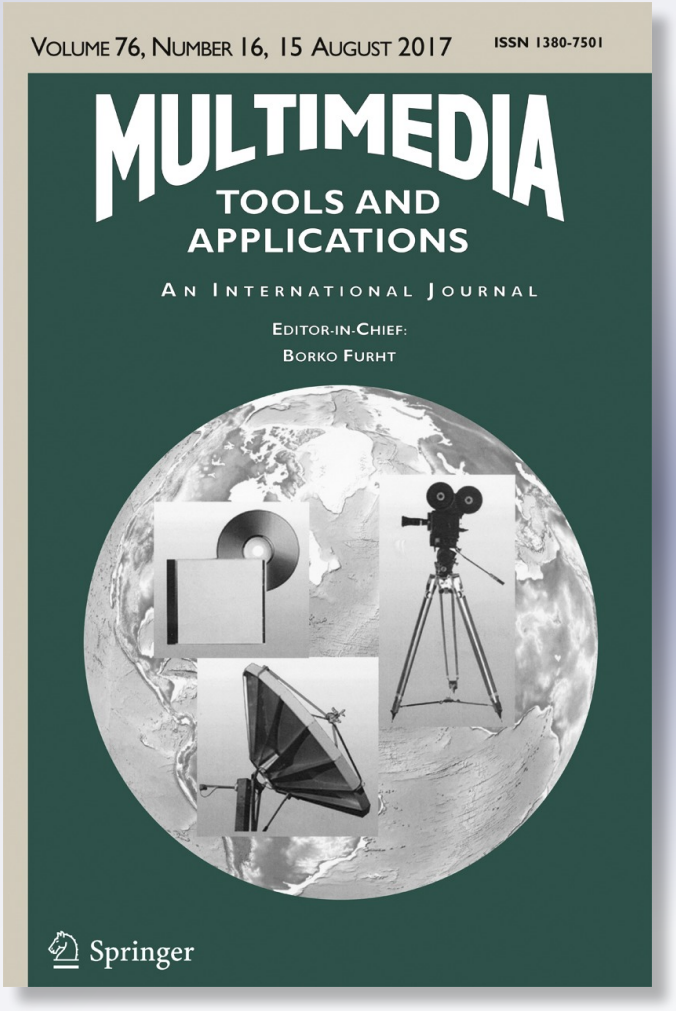
Parallel HEVC decoding with asymmetric mobile multicores

Seehwan Yoo & Eun-Seok Ryu

Multimedia Tools and Applications
An International Journal

ISSN 1380-7501
Volume 76
Number 16

Multimed Tools Appl (2017)
76:17337-17352
DOI 10.1007/s11042-016-4269-2



Springer

Multimed Tools Appl (2017) 76:17337–17352
DOI 10.1007/s11042-016-4269-2

Parallel HEVC decoding with asymmetric mobile multicores

Seehwan Yoo[1] · **Eun-Seok Ryu**[2]

Received: 28 July 2016 / Revised: 24 October 2016 / Accepted: 14 December 2016 /
Published online: 7 January 2017

Abstract Recently, the necessity of parallel ultra-high definition (UHD) video processing has been emerging, and the usage of the computing systems that have asymmetric processors, such as ARM big.LITTLE, is actively increasing. Thus, a new parallel UHD video processing method optimized for asymmetric multicore systems is essential. This paper proposes a novel High Efficiency Video Coding (HEVC) Tile partitioning method for the parallel processing by analyzing the computational power of asymmetric multicores. (1) The proposed method analyzes the computing power of asymmetric multicores and (2) the regression model of computational complexity per video resolution. Lastly, (3) the model determines the optimal HEVC Tile resolution for each core and partitions and allocates the Tiles to suitable cores. Experimental results with the test sequences of common test condition (CTC) show that the decoding speed improved by 17 % with implemented multi-threading module on ARM asymmetric multicore systems.

Keywords HEVC · Parallel video processing · Mobile processor · Asymmetric multicore

1 Introduction

With the advent of wider, clearer display screen, UHD video draws big attention even in recent mobile devices. Larger screen with higher resolution dramatically enhances user experiences at the cost of more pixels processing. 4K UHD video screen has 4 times

✉ Eun-Seok Ryu
esryu@gachon.ac.kr

Seehwan Yoo
seehwan.yoo@dankook.ac.kr

[1] Dankook University, 152, Jukjeon-ro, Suji-gu, Yongin-si, Gyeonggi-do 16890, South Korea

[2] Gachon University, 1342, Seongnam-daero, Sujeong-gu, Seongnam-si, Gyeonggi-do 13120, South Korea

more pixels than Full HD (FHD) screen. Namely, a UHD video picture demands more than 4 times pixels processing capacity, compared with Full HD. Despite the dramatic enhancements of user experiences from large screen, 4K UHD video is one of the most computing-intensive applications within a mobile device.

Multicore utilization definitely helps UHD video processing. Due to the increased performance demand, HEVC (High efficiency video coding) standard defines parallel-friendly encoding/decoding schemes, such as Wavefront parallel processing (WPP), Tiles, and Slides [12]. In such schemes, a screen is divided into multiple processing regions (Tiles and slides), and each parallel-processing core concurrently encodes/decodes the video data only for its designated region. Tile makes video processing parallel-friendly, minimizing sequential fraction of execution, and make easy to leverage recent mobile multicore CPUs.

Yet, a concern for UHD video processing in mobile devices is limited battery lifetime because a mobile device runs with physically constrained battery capacity. Due to the increased screen size, we need to process more pixels, intensively consuming physically constrained energy budget. Thus, energy-efficient video processing is an essential demand for realizing UHD video processing in mobile devices.

To address both high computing performance such as UHD video processing, and high energy-efficiency for limited battery capacity, recent mobile processors adopt diverse power-performance scaling schemes (e.g. DVFS, DPM, energy-aware scheduling, etc.). One of radical approaches is SI-HMP (Single-ISA, Heterogeneous Multi-core Processor) [25]. SI-HMP consists of heterogeneous multiple cores that are differently optimized for power-performance.

For example, ARM's big.LITTLE mobile processors consist of big and little cores: a big core presents higher computing performance, consuming more power than a little core. A little core presents much less power-consumption with limited computing performance. According to the computing performance demands, or power budget, we can dynamically choose cores-combination because different cores combinations present different power-performance scaling.

To use asymmetric multi-core (or SI-HMP) with UHD video, multi-core utilization is primary consideration in order to meet the performance demands for UHD video processing. Particularly for mobile UHD video processing, performance is the primary objective because the device always interacts with the user. In this regard, an important observation with big.LITTLE SI-HMP is the performance asymmetry between big and little cores. If we provide the same workload to big and little cores, a big core will finish the job much earlier than a little core.

This asymmetry makes the existing parallel-friendly UHD video processing sub-optimal, not only in terms of parallel processing performance, but in terms of energy-efficiency. It will fail to maximize performance due to the bottleneck in a video picture (frame) processing among cores, which would be the slowest little cores; in addition, it will fail to minimize power-consumption because it shall not turn-off big cores.

To adequately leverage multi-core performance and power-efficiency with SI-HMP, UHD video processing should carefully look into performance characteristics of the video processing at two different layers: SI-HMP hardware performance asymmetry, and video codec's parallel processing. Video encoding/decoding should be properly segmented so that it can deliver the asymmetric workload, rather than symmetric, considering to the designated core's computing capacity.

A goal of this paper is to present a cross-layer optimization in mobile video processing. To realize UHD video processing with asymmetric mobile multi-core such as SI-HMP, we estimate each core's processing power at runtime. In addition, we divide the screen into

regions of different computing complexity. Then, we map the region into cores so that all regions can be encoded/decoded in parallel.

The rest of paper is organized as follows: Section 2 comparatively explains related studies. Section 3 motivates our study by presenting the performance asymmetry in SI- HMP cores. Next, Section 4 presents our new video coding with cross-layer optimizations, maximizing parallelism and power-efficiency with preliminary results with our new coding scheme. Finally, Section 5 concludes our paper, and presents some possible future work.

2 Related work

2.1 Power-performance scaling in multicores

Performance-energy efficiency of multi-core has shown theoretically in the literature [20, 23, 39]. In the paper, the authors showed that a combination of one (high performance core, or big core) and multiple (low power core, or small cores) is more energy-efficient than all small or all big cores. In 2003, single-ISA (instruction set architecture), heterogeneous multi-core has been proposed in [25]. It draws attention because it premises of high performance from multi-core and power-efficiency from small cores. In the proposed architecture, all cores share the instruction set, so that software can freely migrate over different cores without any re-compilation.

There are several studies that analyze the power/energy consumption and performance of microprocessor [2, 9, 15–18, 31, 41] in various aspects, from hardware materials technology/die size to software concurrency. Cassidy and Andreou [9] and Esmaeilzadeh et al. [17] pointed out that software concurrency affects the power-performance characteristics. Le Sueur and Heiser [27], Göddeke et al. [21] and Pagani et al. [32] showed power-management through DVFS (Dynamic Voltage and Frequency Scaling) has significant impact on power consumption. Impact from semiconductor technology and physical die size has also been analyzed in [2, 41]. Esmaeilzadeh et al. [15, 16, 18] addressed application characteristics that affect the power-performance scaling.

Recently, [6] comparatively present power-performance characteristics between ARM and Intel CPUs. Although they use different semiconductor technology, ARM CPUs are optimized for low-power range, and Intel CPUs are optimized for high-performance range.

Another recent study [8] presents a unified power-performance control framework for DVFS and multi-core. DVFS controls power consumption along with performance level by throttling CPU's clock speed and input voltage. In addition, multi-core utilization also can increase performance level by utilizing additional core. In the framework, the authors reveal that static power consumption is considerable in mobile CPUs, and manage DVFS and multi-core according to the workload demand.

Aside from the hardware power-performance characteristics, the actual core multi-core utilization is closely related with the OS scheduler. Thus, several scheduler optimizations considering energy-efficiency have proposed in the literature [4, 5, 29, 30, 33–35, 37, 40]. Most of them are based upon the workload profiling. Some of them [4, 5, 37] use on-line profiling to identify the current workload levels. Some of them use [30, 34, 35, 40] off-line profiling to accurately identify the application behavior. Some of them uses micro-architecture level profile information such as the distance between the communicating cores [10], Last-level cache (LLC) information, IPC and L2 cache misses [19]. A recent study proposed an accurate model [37] that identifies the workload characteristics based upon profiled information.

Shelepov et al. [35] and Koufaty et al. [24] pointed out that multi-core CPUs can presents asymmetric performance because of the per-core DVFS policy, or micro-architectural differences. This performance asymmetry affects fairness in the operating systems scheduler. To mitigate the performance asymmetry, age-based scheduler [26], progress fair scheduler [13], profile-based fair scheduler [11] have been proposed.

2.2 HEVC standard for UHD video processing

HEVC the newest video coding standard was standardized by JCT-VC organized group in partnership with ITU-T VCEG (Video Coding Experts Group) [7, 36] and ISO/IEC MPEG (Moving Picture Experts Group). HEVC is based on block-based hybrid video approach like existing major video coding standards. HEVC provides advanced video compression capability to support UHD videos such as 4K, 8K by providing new parallel processing tools (Tile and WPP.)

2.2.1 Encoding/decoding complexity prediction algorithms

Decoding complexity of video frames is affected by many explicit factors (e.g. resolution, QP) and others. Recently, the researches on predicting encoding/decoding complexity are proceeding to optimize power efficiency and encoding/decoding time.

One of them adjusts clock speed of CPU to save power. It sets low clock speed when frame will be decoded has less computational complexity and sets high clock speed when a frame has much computational complexity. This research is meaningful as aspects to improve power efficiency but it do not consider parallelism on multicore systems and optimization of decoding time. Other research is Tile partitioning algorithm based on the number of bits of CTUs (Coding Tree Unit) [3]. The algorithm propose the method to

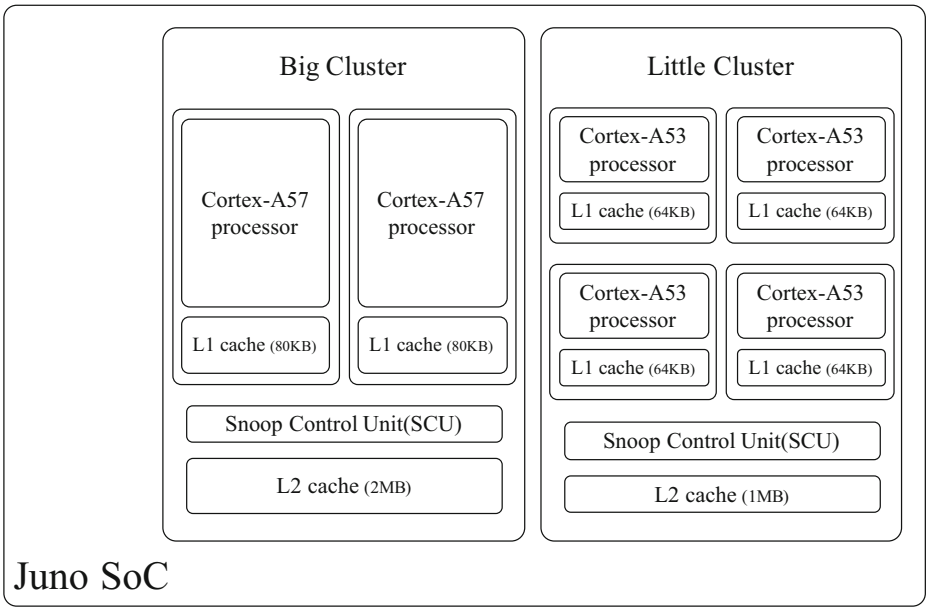


Fig. 1 Single-ISA asymmetric multi-core processor, Juno CPU

Table 1 Juno's big/little core u-arch comparison

	little core (Cortex-A53)	big core (Cortex-A57)
pipeline	in-order	out-of-order
issue width	dual-issue	8-issue, 3-wide decoder
L2 cache	1MB	2MB, 16-way set assoc., banked
TLB	512 entry, 10 entry uTLB	1024 entry, 48 entry u-ITLB, 32 entry u-DTLB
branch prediction	4Kb conditional, 256 entry indirect predictor	4K BTB, indirect with path history predictor
power-saving features	hierarchical clock gating, power domains, advanced retention mode	tag-reduction, way-prediction, cache-lookup suppression

equalize the total number of bits in each Tiles in order to minimize decoding time between Tiles have a lot of bits or a few bits. It has many similarities with our research, but the research uses different method to predict complexity and do not consider asymmetric multicore systems.

3 Power-performance characteristics of SI-HMP

Video processing is one of processing intensive applications even with mobile devices. To meet the performance demand of high quality, high resolution video processing, as well as energy-efficiency of battery-limited mobile devices, we should carefully investigate the underlying multicore performance characteristics.

In recent mobile multi-cores, there are several options to leverage CPU performance and power consumption. At first, we increase performance by utilizing multiple cores. Secondly, we can use DVFS (Dynamic Voltage and Frequency Scaling), throttling core speed along with the voltage. Operating performance points (OPP) defines CPU's core clock speed and operating input voltage. Thirdly, we can use heterogeneous micro-architecture cores. For example, ARM's big.LITTLE CPU consists of different kinds of multiple cores. One variant

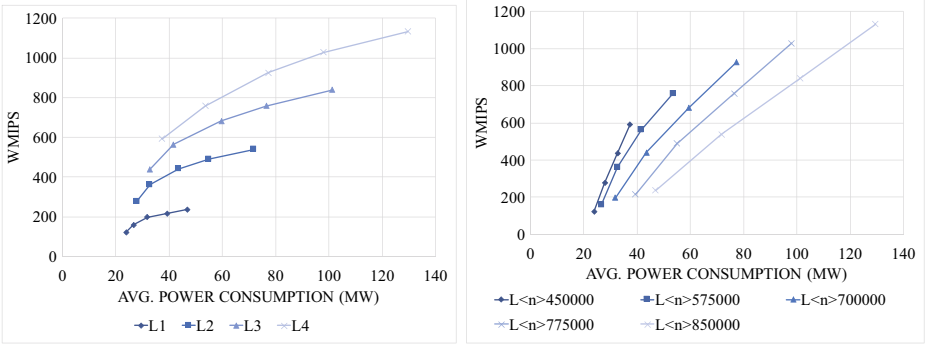


Fig. 2 Little cores' power-performance with WhetstoneMP

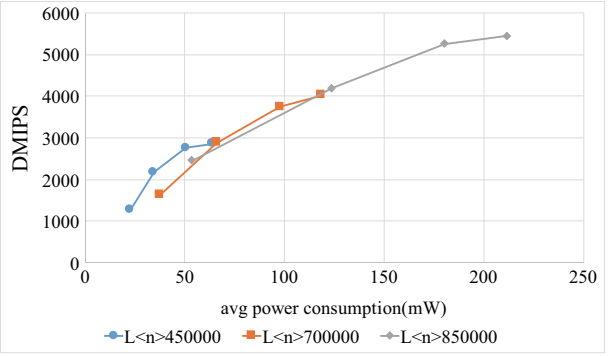


Fig. 3 Little cores' power-performance with DhrystoneMP: multicore scaling

of SI-HMP is SI-AMP, (single-ISA asymmetric multicore processor). An example of SI-AMP is ARM's Juno CPU [1], that consists of dual big cores (Cortex-A57 cores), and quad little cores (Cortex-A53 cores) as shown in Fig. 1. Hardware specification of big and little cores of Juno CPU is presented in Table 1.

The asymmetry in power and performance of big and little cores, as well as the numbers of big and little cores, pose a significant challenge in parallel video processing because existing parallel methods assume symmetric processing power among all cores.

In this section, we present performance asymmetry in Juno CPU in order to clearly present the underlying hardware performance characteristics.

3.1 Power-performance of Little cores cluster

To understand underlying hardware power-performance characteristics, we run some CPU-intensive benchmarks in various configurations. At first, we focus on the performance and power consumption of little cores cluster. In our Juno CPU, there are four little cores, and each core defines five operable frequencies as follows: 450 Mhz, 575 Mhz, 700 Mhz, 775 Mhz, 800 Mhz. We run WhetstoneMP [38], and measure throughput and average power consumption during the effective execution, changing the operating frequency, and the active number of cores.

Figure 2a shows the power-performance scaling of little cores cluster. X-axis is the average power consumption in milliWatt (mW), and Y-axis is the achieved throughput (Whetstone Million Instructions Per Second, WMIPS).

Table 2 Stall cycles in DhrystoneMP execution

Threads	Load miss stall	Store stall	Stall cycles in $core_0$	Total cycles in all cores	Stall cycles %
1	4,293,908	6,281,047	10,574,955	396,223,418	2.669 %
2	191,242,001	28,808,093	220,050,094	963,227,230	22.845 %
3	603,522,824	129,965,692	733,488,516	1,818,333,733	40.338 %
4	1,444,624,241	246,052,199	1,690,676,440	3,146,397,125	53.734 %

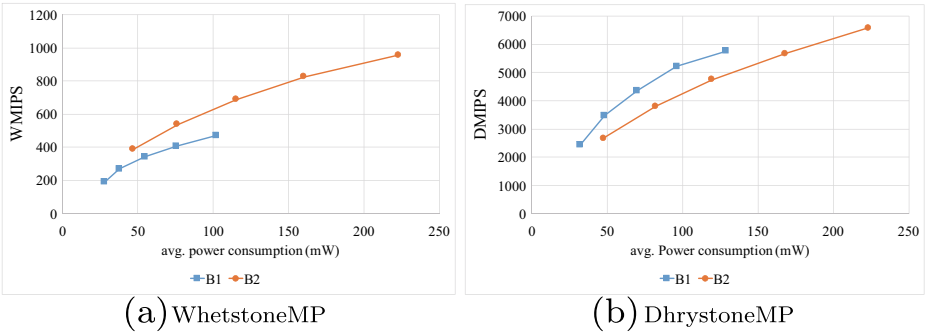


Fig. 4 Big cores' power-performance

In the graph, the same line presents the frequency scaling. In the graph, the gradient of curve decreases gradually, meaning that performance gain per additional power decreases. In short, energy-efficiency decreases as we increase performance.

Compared against frequency scaling, multi-core scaling is more preferable when power consumption is in reasonable range. Figure 2b is the same graph with different presentation. In the figure, each line presents (power, performance) for the same frequency, changing the number of utilizing cores. Comparing two figures, the slope with multi-core scaling is much steeper than frequency scaling, meaning that multi-core scaling is more power-efficient than frequency scaling.

For different workload, performance scaling with multicore can be limited by Amdahl's Law. For DhrystoneMP [14], we get different result. Figure 3 shows the average power consumption and achieved performance of DhrystoneMP.

Differently from Whetstone, most of the dots in Fig. 3 are on the same line, meaning that frequency scaling and multicore scaling presents similar power-performance scaling. A reason of the different results between DhrystoneMP and WhetstoneMP is that multicore scaling is sensitive to workload characteristics.

Table 2 shows the stall cycles and total instruction execution cycles according to the number of active cores (threads). In the table, as the number of thread increases, the percentage of stall cycles increases as much as 53 % in quad core case. That is, 53 % of CPU cycles are wasted due to cache coherency.

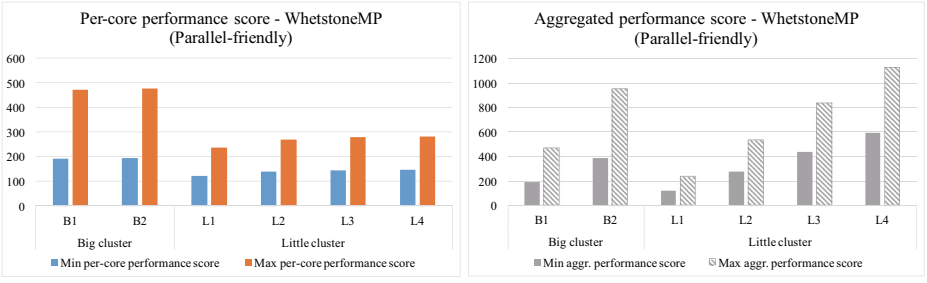


Fig. 5 Per-core performance with parallel-friendly benchmark in SI-AMP

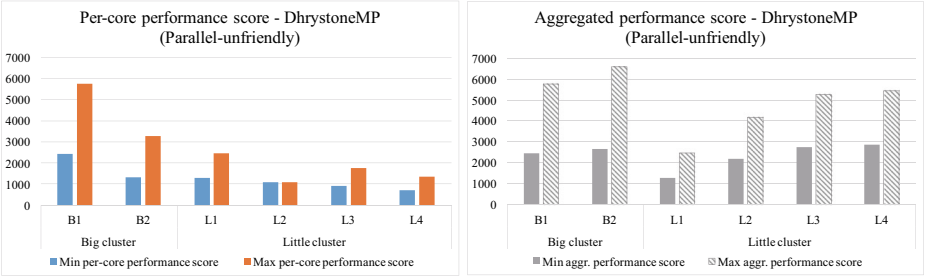


Fig. 6 Per-core performance with parallel-unfriendly benchmark in SI-AMP

In DhrystoneMP implementation, there are global data that is shared among the different threads, and the shared data is frequently updated by multiple cores, increasing the cache miss rates, which hinders parallel execution, and reduces parallel execution fraction in Amdahl's law. Namely, if the workload is not parallel-friendly, the performance gain from multicore can be limited.

3.2 Power-performance of Big cores cluster

Juno CPU has two big cores, aiming at high power performance range. It uses more deep and wide issue pipeline structure, larger TLB/cache than little cores. Adopting more complex micro-architecture, utilizing wider floor-plan, consuming more transistors, big cores achieve higher throughput than little cores. Similarly to little cores, we measure power and performance of big cores, according to the number of active cores and operating frequencies. Juno's big cores define the following five operating frequencies: 450 Mhz, 625 Mhz, 800 Mhz, 950 Mhz, 1.1 Ghz.

Figure 4a and b shows the results with WhetstoneMP and DhrystoneMP, respectively. Each line presents frequency scaling with fixed number of active core. Note that X-axis scale

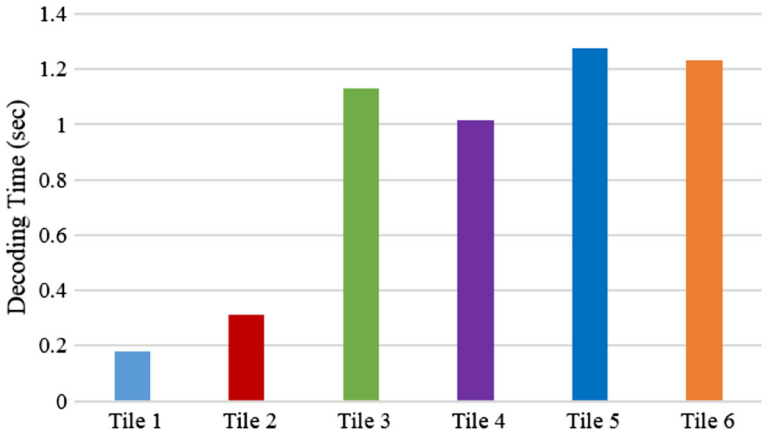


Fig. 7 The average decoding time of each Tile (Sequence: PeopleOnStreet)

is different from previous figures. In the graph, the curve shows more gradual performance gain than little cores cluster. That means, big cluster is less power-efficient than little cluster.

From the performance result of DhrystoneMP in Fig. 4b, we can see that additional core does not enhance performance much. With additional core, performance enhancement is only about 15 %. A reason is that given workload is not parallel-friendly.

3.3 Asymmetry in power-performance of big-little clusters

From the benchmark result, we see that performance scaling with multicore can be beneficial especially when the workload is parallel-friendly. For parallel-friendly benchmark, WhetstoneMP, performance enhancement by additional core is very steep. Therefore, little cores cluster shows higher whetstone MIPS (WMIPS) score, than big cores cluster, in its peak point.

(a) Uniform Tiles partitioning

(b) Non-uniform Tiles partitioning for 2 big cores and 4 little cores

Fig. 8 Uniformly and not uniformly divided frame into Tiles

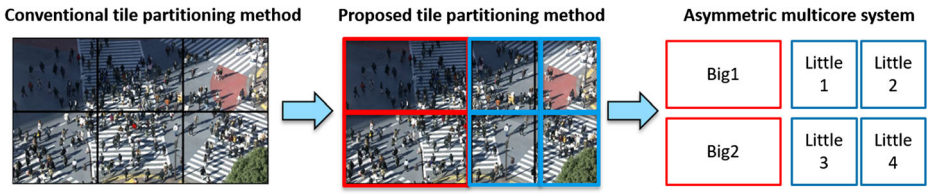


Fig. 9 The conceptual diagram of the proposed Tile allocation method

Figure 5 shows per-core performance scores and aggregated scores from WhetstoneMP. To comparatively present the per-core performance, we divide the MIPS score by the number of cores. In the figure, per-core score of big cluster is higher than little cluster. Per-core performance in little cluster does not decrease as the number of cores because the workload is parallel-friendly. Therefore, the aggregated performance score of little cluster sharply increases and the little cluster achieves higher score when four cores are used at the same time. Note that when the workload is parallel-friendly, per-core performance remains almost at the same level.

For another workload, DhrystoneMP, which is parallel-unfriendly workload, results are slightly different. Figure 6 shows per-core and aggregated performance score from DhrystoneMP. In the figure, per-core performance score decreases rapidly, differently from Fig. 5. If the workload is not parallel-friendly, additional cores are not efficiently utilized, and aggregated performance remains at the same level, as shown in the figure.

4 Parallel video processing with asymmetric performance cores

This section describes that video systems on asymmetric platform need an optimized parallel processing method for asymmetric multicore. Furthermore, this paper propose s novel

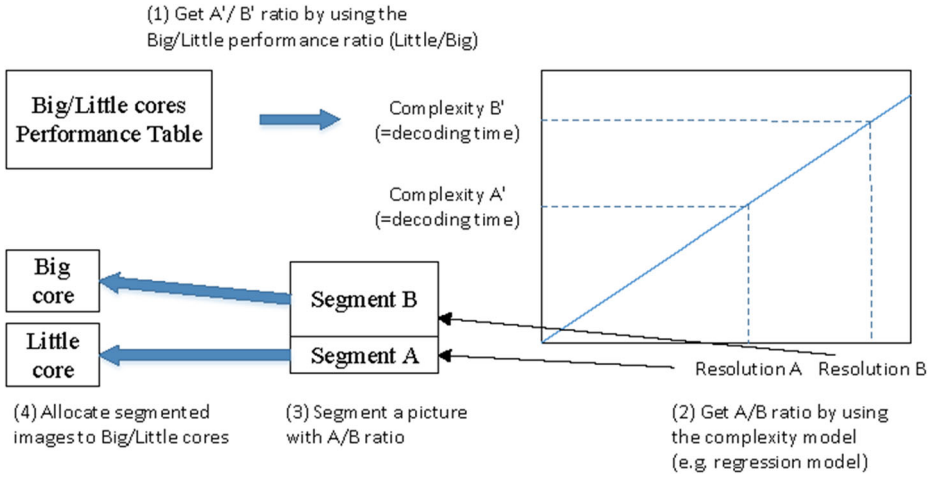


Fig. 10 The procedure of proposed method

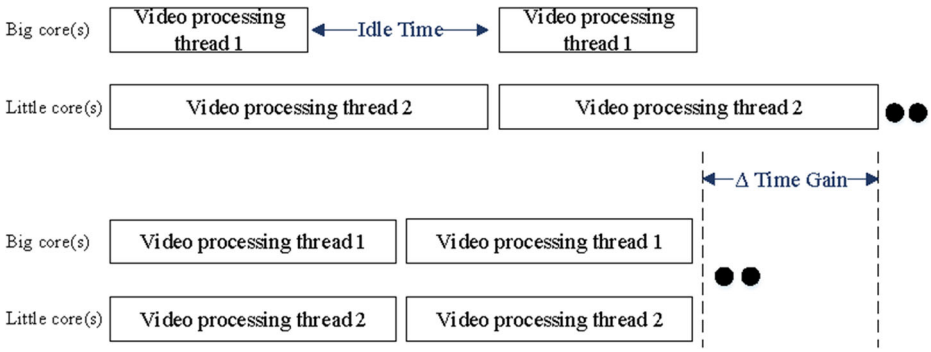


Fig. 11 Expecting decoding time gain from considering asymmetric performance ratio of big/little cores

Tile partitioning method minimizes the gap of the decoding time between the fastest CPU core and the slowest one.

4.1 Problem of uniform tile partitioning

Figure 7 shows an example of decoding time of PeopleOnStreet sequence split into 6 Tiles uniformly as shown in Fig. 8a. It shows the significant difference of decoding times of Tile1, Tile2, and others. In the example, Tile1 and Tile2 are allocated to big cores and other Tiles are allocated to little cores. The difference of decoding times is caused by ignoring the decoding complexity and the computing ability of allocated core. Thus, the thread for the Tile that has the shortest decoding time needs to wait for the slowest thread even if the fastest thread completed its processing.

This paper proposes Tile partitioning method based on the resolution of Tiles and the performance ratio of big core and little core as shown in Fig. 9. It optimizes decoding time of a video sequence by partitioning/allocating non-uniform Tiles to suitable cores.

4.2 Tile partitioning-based HEVC parallel decoding for asymmetric multicore processor

The proposed method is based on the regression model indicating a correlation between resolutions and decoding complexity. Figure 10 shows the procedure that (1) calculates the ratio between the decoding complexities of A' and B' based on the performance ratio of big and little cores, (2) obtains resolutions of Tiles by using a complexity–resolution regression model, (3) segments a picture into non-uniform Tiles, and (4) allocates segmented Tiles to big and little cores.

Table 3 Experiment environments

Big cluster	Cortex-A57 r0p0
LITTLE cluster	Cortex-A53 r0p0
Number of Cores	2 Big cores (1.1 GHz)
	4 Little cores (850 MHz)
Memory	8GB DDR RAM

Table 4 Coding options for test sequences

Coding Option	Parameter
Coding Structure	RA (Random Access)
	AI (All Intra)
	LDB (Low-Delay B)
QP	22, 27, 32, 37
Number of Tiles	6 (3 × 2)

The proposed non-uniform Tile partitioning method makes decoding time gain by equalizing decoding time of each thread, and it is as shown in Fig. 11.

4.3 Performance verification

In this paper, we use HM15.0 (HEVC Reference Model) [22] to encode/decode test sequences on JUNO ARM Development Platform asymmetric multicore platform using ARM big.LITTLE. Also, decoding functions in HM15.0 are modified to allocate threads to suitable core using libde265 software [28].

Table 3 presents a hardware specification of experiment environments. During the experiments, two test sequences such as PeopleOnStreet (3840 × 2160, 150 Frames), Traffic (3840 × 2048, 300 Frames) are employed according to the CTC and are coded as shown in Table 4. The CTC is defined in JCT-VC for HEVC standard work and recommend four QP values such as 22, 27, 32, and 37.

Firstly, this study simulates the performance of parallel processing using the decoding time summation of each single process. Thus, Table 5 and Fig. 12 show the simulated results. The test results show that the proposed Tile partitioning method achieves decoding time gain of average 18.8 % compared to uniform Tile partitioning method. Secondly, this study implemented multi-threading modules for real video decoding tests, and achieved decoding time gain was up to 17.35 %.

Table 5 Decoding time gains using proposed method

Test Sequence	Resolution	Frame length	Number of Tiles	QP	Decoding Time Gain (%)		
					RA	AI	LDB
PeopleOnStreet	3840 × 2160	150	6 (3×2)	22	28.11	25.81	21.80
				27	19.05	9.20	11.77
				32	17.83	14.31	07.43
				37	07.45	12.53	11.98
Traffic	3840 × 2048	300	6 (3×2)	22	19.55	20.74	23.35
				27	22.31	21.20	19.24
				32	25.06	16.67	19.69
				37	24.72	28.11	23.20

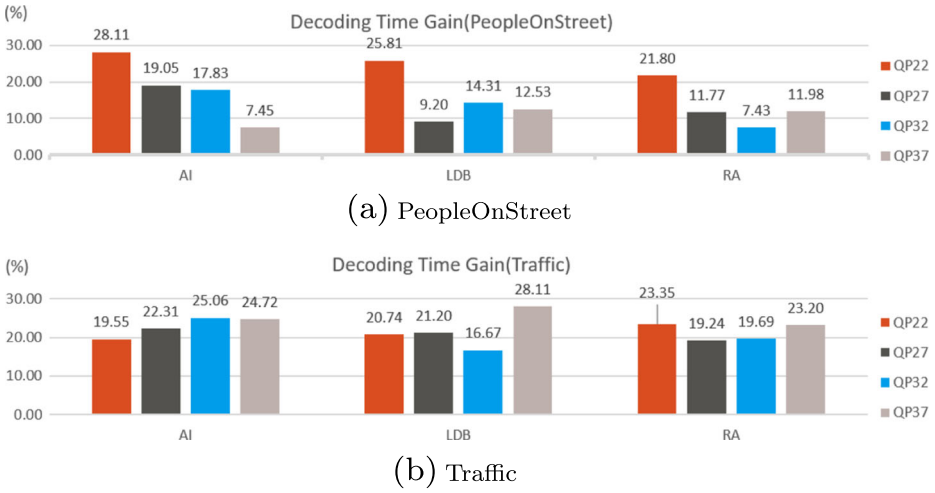


Fig. 12 Decoding time gains per encoding configuration using proposed method

5 Conclusion

This paper explains a new parallel HEVC decoding method with asymmetric mobile multi-cores and proposes the HEVC Tile partitioning method. The proposed method partitions a frame into non-uniform Tiles and allocates threads to suitable cores. It achieves improved decoding time gains which are about 20 % with single process-based simulation and about 17.35 % with implemented multi-threading experiments compared to the uniform Tile partitioning without considering computing power of each core.

Acknowledgments This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2015R1C1A1A02037743 and NRF-2015R1C1A1A02037330), and this research was also partially supported by the Gachon University research fund of 2015.(GCU-2015-0045).

References

1. ARM: Juno arm development platform, Available online: http://www.arm.com/products/tools/development-boards/versatile-express/juno-arm-development-platform.php
2. Azizi O, Mahesri A, Lee BC, Patel SJ, Horowitz M (2010) Energy-performance tradeoffs in processor architecture and circuit design: a marginal cost analysis. In: Proceedings of the 37th annual international symposium on computer architecture, ISCA '10. ACM, New York, pp 26–36. doi:10.1145/1815961.1815967
3. Baik H, Song H (2015) A complexity-based adaptive tile partitioning algorithm for hevc decoder parallelization. In: 2015 IEEE international conference on image processing (ICIP). IEEE, pp 4298–4302
4. Belviranli ME, Bhuyan LN, Gupta R (2013) A dynamic self-scheduling scheme for heterogeneous multiprocessor architectures. ACM Trans Archit Code Optim 9(4):57:1–57:20. doi:10.1145/2400682.2400716
5. Bhadauria M, McKee SA (2010) An approach to resource-aware co-scheduling for cmps. In: Proceedings of the 24th ACM international conference on supercomputing, ICS '10. ACM, New York, pp 189–199. doi:10.1145/1810085.1810113
6. Blem E, Menon J, Sankaralingam K (2013) Power struggles: Revisiting the risc vs. cisc debate on contemporary arm and x86 architectures. In: Proceedings of the 2013 IEEE 19th international symposium on high performance computer architecture (HPCA), HPCA '13. IEEE Computer Society, Washington, DC, pp 1–12. doi:10.1109/HPCA.2013.6522302

7. Bossen F, Bross B, Suhring K, Flynn D (2012) Hevc complexity and implementation analysis. IEEE Trans Circ Syst Vid Technol 22(12):1685–1696
8. Carroll A, Heiser G (2014) Unifying dvfs and offlining in mobile multicores. In: 2014 IEEE 19th real-time and embedded technology and applications symposium (RTAS), pp 287–296. doi:10.1109/RTAS.2014.6926010
9. Cassidy AS, Andreou AG (2012) Beyond amdahl's law: an objective function that links multiprocessor performance gains to delay and energy. IEEE Trans Comput 61(8):1110–1126. doi:10.1109/TC.2011.169
10. Chen J, John LK (2009) Efficient program scheduling for heterogeneous multi-core processors. In: Design automation conference, 2009. DAC '09. 46th ACM/IEEE, pp 927–930
11. Chen Q, Guo M (2014) Adaptive workload-aware task scheduling for single-isa asymmetric multicore architectures. ACM Trans Archit Code Optim 11(1):8:1–8:25. doi:10.1145/2579674
12. Chi CC, Alvarez-Mesa M, Juurlink B, Clare G, Henry F, Pateux S, Schierl T (2012) Parallel scalability and efficiency of hevc parallelization approaches. IEEE Trans Circ Syst Vid Technol 22(12):1827–1838
13. Craeynest KV, Akram S, Heirman W, Jaleel A, Eeckhout L (2013) Fairness-aware scheduling on single-isa heterogeneous multi-cores. In: Proceedings of the 22nd international conference on parallel architectures and compilation techniques, pp 177–187. doi:10.1109/PACT.2013.6618815
14. Dhrystonemp. http://www.roylongbottom.org.uk/android%20multithreading%20benchmarks.htm#anchor6
15. Esmaeilzadeh H, Blem E, St Amant R, Sankaralingam K, Burger D (2011) Dark silicon and the end of multicore scaling. In: Proceedings of the 38th annual international symposium on computer architecture, ISCA '11. ACM, New York, NY, USA, pp 365–376. doi:10.1145/2000064.2000108
16. Esmaeilzadeh H, Blem E, St Amant R, Sankaralingam K, Burger D (2012) Power limitations and dark silicon challenge the future of multicore. ACM Trans Comput Syst 30(3):11:1–11:27. doi:10.1145/2324876.2324879
17. Esmaeilzadeh H, Cao T, Xi Y, Blackburn, SM McKinley KS (2011) Looking back on the language and hardware revolutions: Measured power, performance, and scaling. In: Proceedings of the sixteenth international conference on architectural support for programming languages and operating systems, ASPLOS XVI. ACM, New York, NY, USA, pp 319–332. doi:10.1145/1950365.1950402
18. Esmaeilzadeh H, Cao T, Yang X, Blackburn SM, McKinley KS (2012) What is happening to power, performance, and software? IEEE Micro 32(3):110–121. doi:10.1109/MM.2012.20
19. Ghiasi S, Keller T, Rawson F (2005) Scheduling for heterogeneous processors in server systems. In: Proceedings of the 2nd conference on computing frontiers, CF '05. ACM, New York, NY, USA, pp 199–210. doi:10.1145/1062261.1062295
20. Govindan MSS, Robatmili B, Li D, Maher BA, Smith A, Keckler SW, Burger D (2014) Scaling power and performance viaprocessor composability. IEEE Trans Comput 63(8):2025–2038. doi:10.1109/TC.2013.48
21. Göddeke D, Komatitsch D, Geveler M, Ribbrock D, Rajovic N, Puzovic N, Ramirez A (2013) Energy efficiency vs. performance of the numerical solution of pdes: an application study on a low-power arm-based cluster. J Comput Phys 237:132–150. doi:10.1016/j.jcp.2012.11.031. http://www.sciencedirect.com/science/article/pii/S0021999112007115
22. HEVC Software, Available online: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/
23. Hill MD, Marty MR (2008) Amdahl's law in the multicore era. Computer 41(7):33–38. doi:10.1109/MC.2008.209
24. Koufaty D, Reddy D, Hahn S (2010) Bias scheduling in heterogeneous multi-core architectures. In: Proceedings of the 5th european conference on computer systems, EuroSys '10. ACM, New York, NY, USA, pp 125–138. doi:10.1145/1755913.1755928
25. Kumar R, Farkas KI, Jouppi NP, Ranganathan P, Tullsen DM (2003) Single-isa heterogeneous multi-core architectures: The potential for processor power reduction. In: Proceedings of the 36th annual IEEE/ACM international symposium on microarchitecture, MICRO 36. IEEE Computer Society, Washington, DC, USA, p 81. http://dl.acm.org/citation.cfm?id=956417.956569
26. Lakshminarayana NB, Lee J, Kim H (2009) Age based scheduling for asymmetric multiprocessors. In: Proceedings of the conference on high performance computing networking, storage and analysis, SC '09. ACM, New York, NY, USA, pp 25:1–25:12. doi:10.1145/1654059.1654085
27. Le Sueur E, Heiser G (2010) Dynamic voltage and frequency scaling: The laws of diminishing returns. In: Proceedings of the 2010 international conference on power aware computing and systems, HotPower'10. USENIX Association, Berkeley, CA, USA, pp 1–8. http://dl.acm.org/citation.cfm?id=1924920.1924921
28. libde265, Available online: http://www.libde265.org/
29. Lin FX, Wang Z, Zhong L (2014) K2: A mobile operating system for heterogeneous coherence domains. In: Proceedings of the 19th international conference on architectural support for

programming languages and operating systems, ASPLOS '14. ACM, New York, NY, USA, pp 285–300. doi:10.1145/2541940.2541975
30. Lukefahr A, Padmanabha S, Das R, Dreslinski R, Wenisch TF, Mahlke S (2014) Heterogeneous microarchitectures trump voltage scaling for low-power cores. In: Proceedings of the 23rd international conference on parallel architectures and compilation, PACT '14. ACM, New York, NY, USA, pp 237–250. doi:10.1145/2628071.2628078
31. Morad TY, Weiser UC, Kolodny A, Valero M, Ayguad E (2006) Performance, power efficiency and scalability of asymmetric cluster chip multiprocessors. IEEE Comput Archit Lett 5(1):14–17. doi:10.1109/L-CA.2006.6
32. Pagani S, Chen JJ, Li M (2015) Energy efficiency on multi-core architectures with multiple voltage islands. IEEE Trans Parallel Distrib Syst 26(6):1608–1621. doi:10.1109/TPDS.2014.2323260
33. Pricopi M, Mitra T (2014) Task scheduling on adaptive multi-core. IEEE Trans Comput 63(10):2590–2603. doi:10.1109/TC.2013.115
34. Pricopi M, Muthukaruppan TS, Venkataramani V, Mitra T, Vishin S (2013) Power-performance modeling on asymmetric multi-cores. In: Proceedings of the 2013 international conference on compilers, architectures and synthesis for embedded systems, CASES '13. IEEE Press, Piscataway, NJ, USA, pp 15:1–15:10. http://dl.acm.org/citation.cfm?id=2555729.2555744
35. Shelepov D, Saez Alcaide JC, Jeffery S, Fedorova A, Perez N, Huang ZF, Blagodurov S, Kumar V (2009) Hass: a scheduler for heterogeneous multicore systems. SIGOPS Oper Syst Rev 43(2):66–75. doi:10.1145/1531793.1531804
36. Sullivan GJ, Ohm JR, Han WJ, Wiegand T (2012) Overview of the high efficiency video coding (hevc) standard. IEEE Trans Circ Syst Vid Technol 22(12):1649–1668
37. Van Craeynest K, Jaleel A, Eeckhout L, Narvaez P, Emer J (2012) Scheduling heterogeneous multi-cores through performance impact estimation (pie). In: Proceedings of the 39th annual international symposium on computer architecture, ISCA '12. IEEE Computer Society, Washington, DC, USA, pp 213–224. http://dl.acm.org/citation.cfm?id=2337159.2337184
38. Whetstonemp. http://www.roylongbottom.org.uk/linux%20multithreading%20benchmarks.htm#anchor3
39. Woo DH, Lee HHS (2008) Extending amdahl's law for energy-efficient computing in the many-core era. Computer 41(12):24–31. doi:10.1109/MC.2008.494
40. Zhu Y, Reddi VJ (2013) High-performance and energy-efficient mobile web browsing on big/little systems. In: Proceedings of the 2013 IEEE 19th international symposium on high performance computer architecture (HPCA), HPCA '13. IEEE Computer Society, Washington, DC, USA, pp 13–24. doi:10.1109/HPCA.2013.6522303
41. Zidenberg T, Keslassy I, Weiser U (2012) Multiamdahl: How should i divide my heterogenous chip? IEEE Comput Archit Lett 11(2):65–68. doi:10.1109/L-CA.2012.3

Seehwan Yoo is an assistant professor at the department of mobile systems engineering in Dankook University, South Korea. Before joining the Dankook University, Dr. Yoo worked for LG electronics as a senior researcher. received Ph.D. degree in computer science and engineering from Korea University, Korea, 2013. The thesis title is 'Real-time support for Xen-ARM mobile virtualization.' His research interests are in the area of design and implementation of mobile systems, including multicore power-performance scaling, OS scheduling, hardening the OS kernel, I/O performance enhancement, etc. In 2006, Dr. Yoo worked for Microsoft research Asia (located in Beijing, China), as a research intern. He is a member of the IEEE, and IEEE Computer Society, and ACM.

Eun-Seok Ryu is an Assistant Professor at the Department of Computer Engineering in Gachon University, Seongnam, Korea. Prior to joining the University in 2015, he was a Principal Engineer at Samsung Electronics, Suwon, Korea, where he led a multimedia team. He was a Staff Engineer at InterDigital Labs, San Diego, California, USA, from Jan. 2011 to Feb. 2014, where he researched and contributed to next generation video coding standards such as HEVC and SHVC. From Sep. 2008 to Dec. 2010, he was a Postdoctoral Research Fellow at GCATT in the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia USA. In 2008, he was a Research Professor at the research institute for information and communication technology in Korea University, Seoul, Korea. His research interests are in the area of multimedia communications that includes video source coding and wireless mobile systems. He received his B.S., M.S., and Ph.D. in computer science from Korea University in 1999, 2001, and 2008, respectively. He is a Senior Member of the IEEE, IEEE Computer Society, and IEEE Communications Society.