*Journal of Multimedia Information System VOL. 5, NO. 4, December 2018 (pp. 273-276): ISSN 2383-7632(Online)*
http://dx.doi.org/10.9717/JMIS.2018.5.4.273

# Brief Paper:
# Multi-Symbol Binary Arithmetic Coding Algorithm for Improving Throughput in Hardware Implementation

Jin-Sung Kim[1], Eung Sup Kim[2], Kyujoong Lee[1*]

**Abstract:** In video compression standards, the entropy coding is essential to the high performance compression because redundancy of data symbols is removed. Binary arithmetic coding is one of high performance entropy coding methods. However, the dependency between consecutive binary symbols prevents improving the throughput. For the throughput enhancement, a new probability model is proposed for encoding multi-symbols at one time. In the proposed method, multi-symbol encoder is implemented with only adders and shifters, and the multiplication table for interval subdivision of binary arithmetic coding is removed. Compared to the compression ratio of CABAC of H.264/AVC, the performance degradation on average is only 1.4% which is negligible.

**Key Words**: Multi-symbol, Entropy Coding, Binary Arithmetic Coding, CABAC, Hardware Implementation

## I. INTRODUCTION

Video compression standards include entropy coding to reduce redundant data effectively. H.264/AVC widely used for a High Definition (HD) video supports Context-Adaptive Variable Length Coding (CAVLC) and Context-Adaptive Binary Arithmetic Coding (CABAC) [1]. In general, CABAC provides 9~14% higher compression performance than CAVLC [7]. However, it is difficult to enhance the throughput of CABAC even if it is implemented in a specifically designed hardware because of dependency between consecutive binary symbols. For encoder hardware to support real time processing, it is essential to increase the throughput of a CABAC hardware. In this paper, a novel algorithm is proposed for enhancing the throughput of a CABAC hardware.

The operation of CABAC consists of binarization, context-modeling, Binary Arithmetic Coding (BAC) and context-update. In binarization, symbols are converted to a binary sequence, a bin string. Context-modeling selects the best matched context model that is kept up to date by considering the latest encoded symbols. Context model means a probability model for each context symbol. Based on the assigned probability, binary symbols are encoded by interval subdivision and renormalization. After encoding a current binary symbol, current context models are updated according to encoded values. The probability of a current symbol is determined after encoding the previous symbol. This data dependency limits the throughput of CABAC even when it is implemented in a hardware.

In order to improve the throughput of CABAC in hardware implementation, there have been several researches to remove the dependency between consecutive binary symbols. In [2], renormalization is applied not to binary symbols but to a symbol. In [3], renormalization with a symbol proposed in [2] is implemented in hardware. Interval subdivision has more complex dependency between consecutive binary symbols. Based on the analysis of dependency in interval subdivision, pipeline architecture achieves 1 bin/cycle throughput as long as there is no stall by renormalization [4]. In this research, the throughput is enhanced by serializing computing processes for each pipeline stage. However, the critical path length of this method increases, which limits the maximum frequency of the hardware. In [5], it is proposed to process bypass mode symbols and normal mode symbols in parallel. However, its performance is limited when the ratio of normal mode symbols are higher. CABAC in H.264/AVC was presented in [6], in which the multiplication is replaced with a table.

In this paper, the multi-symbol BAC algorithm with a new probability model is proposed for increasing the throughput of BAC hardware. The proposed method removes the multiplication table. The computing

**Manuscript received December 18, 2018 ; Accepted December 23, 2018. (ID No. JMIS-2018-0058)**
Corresponding Author (*): Kyujoong Lee, Dept. of Electronic Engineering, Sun Moon University, Asan, Korea, 82-41-530-2271, kyujoonglee@sunmoon.ac.kr
[1] Dept. of Electronic Engineering, Sun Moon University, Asan, E-mail: jinsungk@sunmoon.ac.kr
[2] Samsung Electronic System LSI Division, Hwaseong, Korea, E-mail: congeal@me.com

operation of the proposed method consists of only adder and shifter, which prevents increasing the process time for the critical path.

The rest of this paper consists of four sections. Section 2 explains how BAC of H.264/AVC works and Section 3 proposes a new probability model and its application to interval subdivision and renormalization. In Section 4, proposed method is evaluated by the experimental result with H.264/AVC. Finally, Section 5 draws the conclusion of this paper.

## II. ANALYSIS OF BAC in H.264/AVC

### 1. Interval Subdivision and Renormlization in BAC

According to the probability of one input binary symbol, BAC divides the interval as shown in Fig.1. In this figure, *Range* denotes the size of the interval, and *Low* denotes the location of the interval. Depending on whether one input binary symbol is Most Probable Symbol (MPS) or Least Probable Symbol (LPS), the method for dividing the interval is determined. This division process is defined as interval subdivision. The new *Range* for LPS is given by the multiplication of the probability of LPS and the *Range*. The new Range for MPS is obtained by using the probability of MPS and the *Range.* This interval division is repeated up to the final binary symbol, and the *Range* and *Low* of the final interval determine encoded bit stream. As interval subdivision is repeated, the size of the interval is decreased. So, it is necessary to keep the size of the interval above a certain level, which is defined as renormalization.

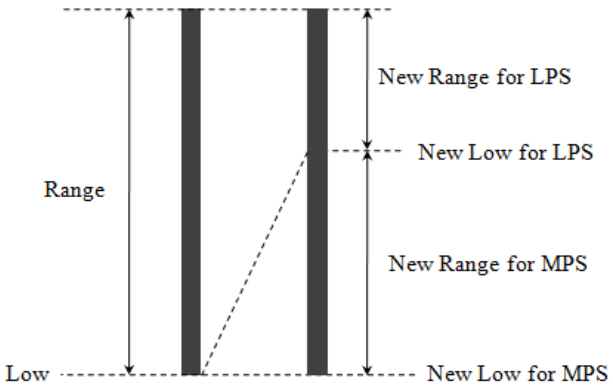


Fig. 1. Interval Subdivision of Binary Arithmetic Coding for LPS and MPS.

In order to calculate a new *Low* for a new input binary symbol, *Range* and *Low* of a current interval are required. For convenience in expression, a *Range* and a *Low* of a current interval are denoted as R and L, respectively, and a *Range* and a *Low* for a new input binary symbol are denoted as R+ and L+, respectively. R+ and L+ are given by Equations (1) and (2), respectively. In both of equations, $R_{LPS}$ stands for the size of a new interval when an input binary symbol corresponds to LPS. $R_{LPS}$ is calculated by multiplying R and the probability for LPS. However, H.264 adopts M-coder to avoid the computation complexity of multiplication. In M-coder, the ranges are quantized and the lookup table is built by multiplying quantized range and the probability for LPS in advance. $R_{LPS}$ is retrieved by utilizing the pre-calculated lookup table. R+ calculated in Equation (1) has four possible values and the number of probability states of LPS is 64. So, the lookup table contains 4x64 pre-calculated values. In Equation (3), the method for retrieving the value from the lookup table in BAC of H.264 is formulized. In this equation, >> stands for right shifts and & means bit-wise AND operation. ρ and σ indicate a quantized range and a state of the probability for LPS, respectively.

$$R+ = \begin{cases} R_{LPS} & for\ LPS \\ R - R_{LPS} & for\ MPS \end{cases} \quad (1)$$

$$L+ = \begin{cases} L + R - R_{LPS} & for\ LPS \\ L & for\ MPS \end{cases} \quad (2)$$

$$R_{LPS} = LookUpTable(\rho, \sigma) \quad (\rho = (R \gg 6)\&3) \quad (3)$$

### 2. Problem Definition for High Throughput BAC

The lookup table in BAC of H.264 eliminates multiplications effectively. However, it limits the throughput of a BAC hardware because of data dependency in a *Range* calculation that requires some serial operations: the quantization of the current interval, the access of the lookup table and renormalization. This serialization prevents parallel computing for consecutive symbols. Thus, the calculation for a current binary symbol cannot start before the calculation for a previous one is complete. For this reason, in order to increase the throughput several times or more, a research for a new *Range* calculation algorithm is needed for the efficient hardware implementation.

## III. MULTI-SYMBOL BAC ALGORITHM

### 1. New Probability Models for LPS

In order to overcome the limitation of the throughput of BAC hardware implementation, new probability models are proposed. It is summarized in Equation (4). In this equation, the probability for LPS is expressed by only the power of 2 such as $2^{-s}$. "s" for indicating the power of 2 is determined

by the probability state, which is the same as the state in the previous method [6]. Proposed probability models require only shift operation. Thus, the multiplication table is not required any more, and the range quantization for the multiplication table is not required as well. The accuracy of the proposed probability model can be lower than that of the previous work in [6] because the number of probability states and the precision of the probability value is low. However, the proposed probability models have no quantization for a range, which can compensate for the performance degradation caused by Equation (4). Experimental results show that the performance degradation in the proposed method is negligible which will be shown in Section 4.

$$R_{LPS} = R \times 2^{-S} \quad (4)$$

**2. Multi-Symbol Algorithm**

In this subsection, the method for implementing multi-symbol BAC with the proposed probability models is proposed. In order to decode an encoded data correctly, entropy coding process of the decoder should be consistent with that of the encoder. For this reason, the bit stream obtained by the parallel computation for multi-symbol should be the same as the bit stream obtained by the serial computation for each single-symbol.

In order to satisfy the consistency, the error caused by truncated precision after encoding the first binary symbol should be compensated for. To this end, floor operation in Equation (5) removes fractional parts for $R_{LPS}$. Since right shift operation removes fractional parts, $R_{LPS}$ is calculated by only right shift operation. For R+ and L+, Equation (5) is applied to Equations (1) and (2), which are calculated by simple additions. After that, renormalization is applied for keeping the size of the range above a certain level. It is described in Equation (6). "m" means the amount of shifts until R+ is above a certain level. "m" is simply retrieved by Leading Zero Detection. Since the range is $R_{LPS}$ for LPS, "m" should be equal to "s" for keeping the size of the range above a certain level. In other words, for LPS, it is the same as fractional bits elimination. It is described in Equation (7).

$$R_{LPS} = floor(R * 2^{-S}) = floor(R \gg s) = R \gg S \quad (5)$$

$$\mathrm{Renorm}(R+) = R \ll m \quad (6)$$

$$\mathrm{Renorm}(R+) = R\&(-1 \ll S) \text{ for LPS} \quad (7)$$

In order to encode $k$ multiple symbols at once, the combination of Equations (1), (2), (5), (6) and (7) is utilized depending on input symbols. In this paper, the combination of equations only for the case of $k$ = 2 is presented and evaluated. However, the same method can be applied to the case when $k$ is greater than 2 and it will still consist of only shift and addition. For the case of $k$ = 2, the possible cases for input symbols are only four cases which are LPS-LPS, LPS-MPS, MPS-LPS and MPS-MPS. The calculations for the *Ranges* of the four cases are summarized in Equation (8) ~ (11). The calculations for the *Lows* of the cases are processed in the same way. R is a current *Range*, and $R_{final}$ is a *Range* that is updated finally for two input bins. R2 means the *Range* when the second input binary symbol is processed. For MPS-MPS, "m1" indicates the amount of shifts for renormalization. Because the probability of MPS is greater than 0.5, "m1" should be equal to 0 or 1. For hardware implementation, it calculates results for m1 = 0 and m1 = 1 in advance and then chooses proper result according to m1 value. In this method, there is no stall to cause the delay.

[LPS-LPS]

$$R_{final} = R \,\&\, \sim(-1 \ll \max(S_1, S_2)) \quad (8)$$

[LPS-MPS]

$$R_{final} = Renorm(R2) \quad (9)$$
$$(R2 = R\&\sim(-1 \ll S_1) - [(R\&\sim(-1 \ll S_1)) \gg S_2)])$$

[MPS-LPS]

$$R_{final} = Renorm(R2) \quad (10)$$
$$(R2 = [R - R \gg S_1]\&\sim(-1 \ll S_2)$$

[MPS-LPS]

$$R_{final} = Renorm(R2) \quad (11)$$
$$(R2 = [(R - R \gg S_1) \ll m1] - [((R - R \gg S_1) \ll m1) \gg S_2]$$

## IV. EVALUATION

In this section, the proposed multi-symbol BAC algorithm is applied to BAC of H.264 and the amount of bits encoded by the proposed method is compared with that encoded by the previous method. For this evaluation, H.264 reference software is modified and the bit consistency between encoder and decoder is confirmed. In Table 1, the sizes of bit stream by CABAC, CAVLC and the proposed method are compared. The amount of bit stream by CABAC is defined as 100%. CAVLC which is less complex than CABAC shows 12.3% increase in bits on average. The proposed method provides only 1.4% increase in bits on average, which is negligible. Even if the proposed method consists of much simpler operations, its performance is very close to CABAC's.

Table 1. The Comparison of Bit Stream Sizes of CABAC, CAVLC and Proposed Algorithm.

| Sequences | CABAC | CAVLC | Proposed algorithm |
|---|---|---|---|
| foreman_cif | 3,543,416 bits | 4,056,576 bits | 3,584,808 bits |
| (300 frames) | 100% | 114.48% | 101.17% |
| tempete_cif | 7,070,592 bits | 7,751,432 bits | 7,163,632 bits |
| (300 frames) | 100% | 109.63% | 101.32% |
| akiyo_qcif | 214,344 bits | 227,360 bits | 216,216 bits |
| (300 frames) | 100% | 106.07% | 100.87% |
| Stockholm_720p | 2,117,920 bits | 2,523,824 bits | 2,142,536 bits |
| (30 frames) | 100% | 119.17% | 101.16% |

## V. CONCULSIONS

In this paper, a novel algorithm to increase the throughput of a BAC hardware is proposed. In proposed algorithm, the probability for LPS is expressed by using only the power of 2 and the interval subdivision is also simplified by the simple probability for LPS. Then, the equations for multi-symbol processing are induced. Those equations consist of only shift and additions. For this reason, the proposed equations are proper for increasing the throughput of the hardware without increasing the critical path length. Furthermore, the degradation of the compression performance is only 1.4%, which shows that proposed algorithm is very effective.

### REFERENCES

[1] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 560–576, July 2003.

[2] D. Marpe, G. Marten, and H. L. Cycon, "A Fast Renormalization Technique for H.264/MPEG4-AVC Arithmetic Coding," *Signal Processing Conference*, 14th European, Sep. 2006.

[3] E.S. Kim, J. Kim, and H.J. Lee, "Binary Arithmetic Coder with One Cycle Data Introduction Interval Designed for H.264 CABAC," *Korean Conference on Semiconductors*, pp.563-564 , Feb. 2009.

[4] C.H. Tsai, Y.J. Chen, and L.G. Chen, "Analysis and architecture design for multi-symbol arithmetic encoder in H.264/AVC," *Proceedings of International SoC Design Conference*, pp.60-63, Oct. 2005.

[5] R.R. Osorio, and J.D. Bruguera, "A new architecture for fast arithmetic coding in H.264 advanced video coder," *Proceedings of 8th Euromicro Conference on DSD*, pp.298-305, Sep. 2005.

[6] D. Marpe, and T. Wiegand, "A highly efficient multiplication-free binary arithmetic coder and its application in video coding," *Proceedings of IEEE International Conference in ICIP*, pp.263-266, Sep. 2003.

[7] D. Marpe, H. Schardwarearz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 620–636, July 2003.