# A Research on Improving of Adaptive Binary Arithmetic Coding Algorithm in H.264

Chongxin Liu [1.a], Wen Dong [2.b]

[1]Hangzhou Normal University, Hangzhou, China

[2]Hangzhou Normal University, Hangzhou, China

[A]lcx860821@163.com, [b]dongwen69@163.com

**Keywords:** Adaptive, Arithmetic, Coding, optimize, h.264

**Abstract.** Adaptive Binary Arithmetic Coding is an effective mode of coding. The paper introduces the basic principle of Adaptive Binary Arithmetic Coding first. Then the paper analyses the complexity of Adaptive Binary Arithmetic Coding and advances a method to optimize the algorithm. Experiment results show that this method can greatly reduce the complexity of Adaptive Binary Arithmetic Coding in h.264.

## Introduction

Arithmetic Coding is one of the main algorithm of image compression. The algorithm is not only a lossless data compression method but also a kind of entropy coding. A basic idea of Arithmetic Coding uses a single floating-point instead of a string of input symbols so that it can obtain a more efficient compression. The method is representing the encoded information as an interval between a real axis of 0 ~ 1 (interval, also called sub-interval). The interval is smaller than longer information, and indicating a longer information needs more number of binary [1]. Arithmetic Coding has a wide application such as text, images, audio, video compression because of its efficient data coding mode.

Ordinary Arithmetic Coding considers the overall statistical properties of the source, but without taking into account the relations between adjacent symbols in the source. In fact, there are strong relations between symbols in many specific applications. In order to improve the efficiency of coding by taking full advantage of the characteristics between adjacent symbols, people proposed the *Context-based Adaptive Binary Arithmetic Coding* (CABAC) [2].

## Principle of Arithmetic Coding

Arithmetic Coding will export a floating-point number $n$ ($0 \le n < 1$), and the input data stream can be obtained by decoding $n$[3].

Assuming that the source has $n$ different symbols $\{ s_1, s_2, ..., s_n \}$, the probability of occurrence of symbol $s_i$ is $P_i$ and $\sum_{i=1}^{n} p_i = 1$. Interval [0,1) is divided into n different sub-interval according to the probability of occurrence of each symbol, i.e. $[0, p_1)$ , $[p_1, p_1 + p_2)$ , ⋯ , $[p_1 + p_2 + \cdots + p_{n-1}, 1)$, then each symbol corresponds to a sub-interval. when the encoder receives a symbol ,the encoder will finds the sub-interval corresponding to it. Then this interval is divided into n different sub-intervals again according to the probability of occurrence of each symbol, each symbol is still corresponds to a sub-interval. Repeat the above process until the encoder receives the last symbol and chose a number from the last sub-interval for output.

## Principle of CABAC

The process of CABAC is mainly divided into three step as Fig. 1 shows [4]:

Published by Atlantis Press, Paris, France.

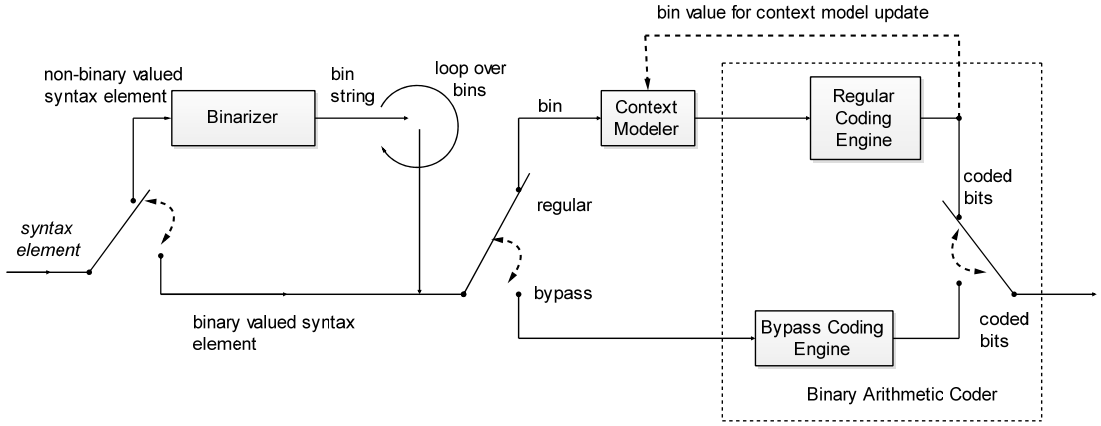


**Fig. 1. CABAC encoder block diagram**

In the first step of the process, a non-binary valued syntax element will be uniquely mapped into a bin-string. If the input is the very binary valued syntax element, the process of *binarization* will be omitted, and the data go to the next step directly through a bypass. Then begin to encode the binary data and there are two coding mode for choice.

In the regular coding mode, every bin of a syntax element will go into the *context-modeler* according to their order which produced by judgments, then the modeler allocates a *probability-model* for the bin according to the coded syntax element or value of the bin, and this process is called *context-modelling*. Then the bin and its probability model was taken together by the regular coding engine. Furthermore, encoder will feedback a message to the context-model according to the value of the bin for context-model update, and this process is called adaptive.

Another mode is bypass coding mode. In this model, no particular probability model allocated for the bin, and the inputted bin is encoded directly by a simple bypass coding engine in order to speed up the process of encoding and decoding. Of course, This mode is only used for some special bin.

## Complexity analysis of Adaptive Binary Arithmetic Coding in H.264.

The H.264/AVC standard is developed by ITU-T and ISO/IEC, positioned to cover the entire video applications, including low rate wireless application, standard-definition and high-definition television broadcast applications, Internet video streaming applications, high-definition DVD video transmission and digital camera with high quality video applications. In this standard, defines two types of entropy coding mode, namely Context-based *Adaptive Binary Arithmetic Coding*(CABAC) and *Context-based Adaptive Variable Length Coding*(CAVLC). H.264 is currently the state-of-the-art video coding standard. Its advanced coding features offer an improvement in the coding efficiency by a factor of about two over MPEG-2 at the expense of higher complexity [5].

After tracking and analyzing each function in CABAC modules, found the function biaries_ncode_symbo1( ) is called the most times. From the flowchart of binary-decision of the H.264 video coding standard can see that encoding a bit need access to at least 4 long forms, i.e. process of CABAC has a large number of operation with memory access, and this is the reason why CABAC has high computational complexity and encoding time.

## Optimization of the CABAC

There is a large number of redundant data exist in actual image data, which comes from the relativity of source image itself and inhomogeneity of probability-distribution, such as correlation of time domain and frequency domain in a image, and the coding efficiency can be optimized by using these redundant information. Choosing the most appropriate context-model can greatly reduce the redundancy between the symbols.

After study the process of initializing probability of the Adaptive Binary Arithmetic Coding context-model we found that the variable CtxState uniquely identifying a context-model, and there

exists a relationship in CtxState and valMPS: va1MPS =CtxState>>6. So just need to use 7bit index of context-model to express arbitrary context-models in 128 models.

On the other hand, this method will make the probability initialization process of the context-model simplified as follows statement: CtxState=Clip3 (L ((m*Clip3, 126, (0, 51, SliceQPY) >>4) +n) . This will greatly optimize the process of initializing probability. This method reduces the complexity of CABAC engine and avoid the turnover problem of valMPS when pStateIdx=0 (pStateIdx is index of probability-state).

In order to validate the correctness of results, Do a test in PC machine with the standard H.264 encoder T264, test sequence level is 5CIF (352 × 288 pixel) format sequence, and they are Others, Newbie, Forest, Taxi, Modem, Congealer. The coding frames were 105, coding rate was 25fps, quantization parameter was 30. The experimental results are as follows:

**Table 1. The comparison of coding performance before and after optimization**

| Test sequences | Others | Newbie | Forest | Taxi | Modem | Congealer |
|---|---|---|---|---|---|---|
| Time of the original context-model code consuming [ $\mu s$ ] | 2734 | 2967 | 2896 | 2789 | 2893 | 3012 |
| Time of the new context-model code consuming [ $\mu s$ ] | 521 | 547 | 520 | 506 | 553 | 564 |
| Rate | 80.94% | 81.57% | 82.05% | 81.87% | 80.88% | 81.26% |

The data from the table above indicate this method can saving about 80% time to the context-model. The optimization make the process of context-model initialization greatly shortened.

## Conclusion

This paper introduces the principle of CABAC and process of encoding in detail, then it analyzes the complexity of CABAC and gives a method to optimize the algorithm. The CABAC algorithm is optimized through choosing a new context-model. The result of the experiment shows that the optimized algorithm can greatly reduce the computational complexity of the CABAC. The algorithm and its hardware, software system structure needs be considered together when CABAC applies in practice. Choosing a good context-model will greatly improve the coding efficiency and save cost.

## Reference

[1] H. Liang, C. Chen, Adaptive Arithmetic Coding and its Application to JPEG2000，Computer & Digital Engineering, 2004, vol. 33, no.7, pp. 56-59.

[2] Y. Li, X. Li, Adaptive Arithmetic Coding and its application in Lossless Image Coding, Computer Engineering and Applications, 2003, vol. 32, pp. 73-75.

[3] Y. Tan, J. Cao, J Tian and J Liu, Compressed Binary-indexed Tree Representing Cumulative Frequency Table for Adaptive Arithmetic Coding, Microelectronics & Computer, 2003, vol. 12, pp. 68-75.

[4]D. Marpe, H. Schwarz, and T. Wiegand, Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard[J], IEEE Transaction Circuits and Systems for Video Technology, 2003, vol. 13, no. 7, pp. 620-636.

[5]N. Hait, D. Malah, Model-Based Transrating of H.264 Coded Vide, IEEE Transaction on Circuits and Systems for Video Technology, 2009, vol. 19, no. 8, pp. 1129-1142.

Published by Atlantis Press, Paris, France.