IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 4, 2020, accepted February 1, 2020, date of publication February 12, 2020, date of current version February 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2973549

Precise Estimation of Probabilities in CABAC Using the Cauchy Optimization Method

DAMIAN KARWOWSKI
Faculty of Computing and Telecommunications, Institute of Multimedia Telecommunications, Poznan University of Technology, 60-965 Poznań, Poland
e-mail: damian.karwowski@put.poznan.pl

This work was supported by the Ministry of Science and Higher Education of Republic of Poland.

ABSTRACT The algorithm of entropy coding that is now widely used in video compression is the Context-based Adaptive Binary Arithmetic Coding (CABAC). This paper presents a modified, improved version of CABAC, called the CABAC+. The author's idea for the improvement is to use in CABAC a more accurate estimation of probabilities of data symbols. The basis of the proposal is to use the Cauchy optimization method, in order to minimize the number of bits that are produced by the entropy encoder. The application of the CABAC improvement in the High Efficiency Video Coding (HEVC) technology increased the compression efficiency of entropy coding by 0.6% to 1.2%, depending on the parameters of the method and scenario of experiments. The use of the proposed solution increases the decoding time of a video, but the method virtually does not change the complexity of a video encoder.

INDEX TERMS Probability estimation, entropy coding and arithmetic coding, improved CABAC, video compression.

I. INTRODUCTION

Direct representation of video samples would result in massive amounts of data. For the case of high resolution video (e.g., 4K video) it would be a huge data stream of 6Gbps or even greater. Therefore, before transmission or storage, video must be compressed. It is performed by a video encoder.

In order to strongly reduce the amount of data, video is described by appropriate syntax elements data, and not directly input samples. A number of data encoding techniques are used for this purpose. One can mention here such techniques as video sample prediction, and transform coding together with data quantization.

Although significant improvements have been achieved this way (in terms of reducing the size of data stream), the syntax elements data still exhibit strong statistical redundancy. In order to reduce this redundancy, entropy coding is always used as the final stage of video compression (see Fig. 1). Entropy coding is lossless coding. However, as practice shows, this compression technique leads to a significant, further increase of efficiency of video compression [1]. Therefore, the importance of this technique in a video encoder is enormous.

The associate editor coordinating the review of this manuscript and approving it for publication was Zhaoqing Pan.

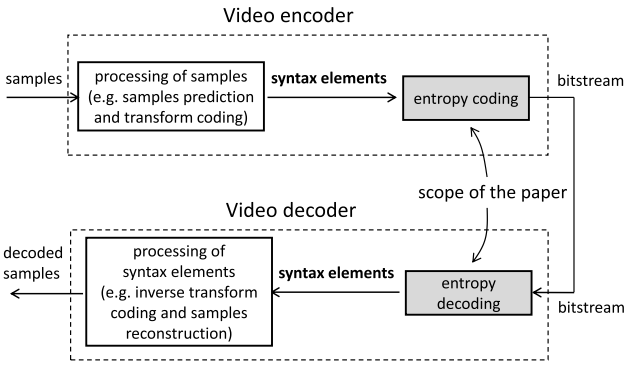

FIGURE 1. A simple block diagram of the video encoder and the video decoder. Entropy coding is of interest to this paper. The scope of the paper are blocks of entropy coding and entropy decoding of the data.

Entropy coding has therefore been the subject of intensive scientific research for years. Recently, these works have mainly been conducted in the context of compression of video data. At present, video data accounts for as much as 80% of all data that is transmitted in telecommunication networks. Therefore, an increase in the efficiency of compressing this type of data even by fractions of percent translates into very large savings of the bandwidth in the scale of total telecommunication traffic. Statistics show the level of importance of such studies. Efficient entropy compression of video data is thus the subject of interest of this work.

The latest video encoders use the technique of arithmetic coding as entropy coding [2]–[5]. Good examples here are two contemporary video coding technologies: 1) H.264/Advanced Video Coding (AVC) [2], [6], which is currently widely used in television and the Internet, and 2) the latest H.265/High Efficiency Video Coding (HEVC) technique [5], [7], which is now the subject of many market implementations worldwide. In those technologies, a very efficient variation of arithmetic coding is used, known in the literature as the Context-based Adaptive Binary Arithmetic Coding (CABAC) technique [8], [9]. It is an arithmetic encoder that ranks among the most efficient entropy encoders practically used in video compression.

In each arithmetic encoder, the key element that affects the encoder performance is the mechanism of estimating the probabilities of the data symbols that are subject to compression. In CABAC, a very sophisticated such mechanism is used (explained more clearly in the next section). It is this mechanism of the entropy encoder that makes the CABAC technique so efficient in data compression.

Despite the high efficiency of the CABAC technique, works are still carried out and described in the literature to further improve the efficiency of arithmetic encoding. These works mainly focus on further improvements in the probability estimation mechanism for coded data symbols. Recently, such studies have been mainly conducted as part of the works of the ISO/IEC Moving Picture Experts Group (MPEG) and the ITU-T Video Coding Experts Group (VCEG) on the Versatile Video Coding (VVC) technology of video compression [10], [11]. In these works, a version of the CABAC algorithm that is used in the HEVC technique (called in this paper 'the HEVC version of CABAC') was used as a starting point [32], [35]. The purpose of this paper is also to develop improvements for the CABAC technique. The goal is to propose a new mechanism for estimating the probabilities of data symbols, which can be used in the CABAC technique. An important element of the paper is also an original proposal on how to incorporate the new mechanism into the CABAC technique. The paper focuses on the type of algorithm improvements that will be usable in the future generations of video encoders.

II. RATIONALE AND GOAL OF THE WORK

A. MAIN ASPECTS OF THE CABAC TECHNIQUE

Data of syntax elements that is the subject of entropy compression in the video encoder is inhomogeneous and very non-stationary. First of all, data of individual syntax elements (for example: motion vectors, coefficients of discrete cosine transformation (DCT), coding modes of image blocks in the hybrid video encoder) have different statistical properties. Secondly, the statistic of syntax elements' data is changing, from frame to frame, and even within a single video frame. Additionally, there are also some relationships between successive binary symbols of the data (inter-symbol correlation), which are very difficult to determine.

In view of the above observations, the estimation of probabilities of syntax elements data is very difficult. It forces

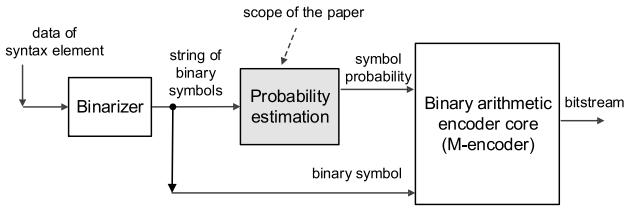

FIGURE 2. A simple block diagram of the CABAC entropy coding technique. The scope of the paper is the block of probability estimation of binary symbols.

the necessity of using complex techniques of probability estimation of data symbols in the entropy encoder and the entropy decoder. The best such example are solutions used in the aforementioned CABAC technique. To describe it very shortly – at the beginning, CABAC represents data of syntax elements with variable-length codes (VLC) when using the binarizer (see Fig. 2). In this way a certain string of binary symbols is obtained. The binary symbols of VLC codes are then appropriately divided into a high number of separate sub-streams (173 independent sub-streams of binary symbols in the HEVC version of CABAC). Finally, the probability of each binary symbol is calculated (in the probability estimation block), when exploiting dependencies that are seen between binary symbols of a sequence [8], [9]. The practical realization of all these stages imposes high burden of computations. Considering the limited computational power of the processors at the time, when the first version of the CABAC algorithm was developed (before the year 2003, for the needs of the AVC video compression technology) one of the CABAC stages has been simplified. It was the stage of calculation of the probabilities of binary symbols (this is the gray block on Fig. 2). Instead of applying some regular method here, a limited set of only 127 predefined probabilities has been created, together with fixed rules for updating the values of probabilities after encoding the 0 or 1 symbol. Thus, when developing the CABAC technique the two significant simplifications have been introduced: 1) related to the number of possible values of probabilities used, and 2) the adopted method of updating the values of probabilities. Those simplifications have, however, allowed CABAC to achieve good balance between the complexity and coding efficiency of the algorithm. Nevertheless, it limited the compression efficiency of the data to some extent. Improvements to the block of probability estimation are of interest to this work.

B. LITERATURE PROPOSALS FOR IMPROVING THE CABAC

Current processors and integrated circuits have, however, much more computing power than those from a dozen years ago. From this point of view, the abovementioned simplifications of CABAC can be canceled today. This gives the opportunity to use in the entropy encoder much more advanced algorithms of estimation of symbol probabilities. Such works have been carried out for years. In those works, efforts were put to calculate the probabilities of binary symbols more precisely than in the original CABAC algorithm.

This goal has been achieved when using several different approaches. And so, in the works [12]–[14] the authors have proposed an extension of the CABAC algorithm that increases the accuracy of symbol probability estimation in the entropy encoder. These are the two-parameter probability update model [12], [13], and the counter-based probability model update [14]. The improvements allowed for an additional increase of compression efficiency of the HEVC version of CABAC by 0.5% to 0.8%. Very similar actions can be observed in the proposal to improve the CABAC algorithm, which is to become part of the currently developed VVC video compression technology [10], [11]. There is talk about an extension of the CABAC probability estimator, which is based on two probability states, instead only one, as in the original algorithm [15]. As the experiments revealed, this proposal allows for an additional increase of coding efficiency by up to 0.7%, when encoding data of syntax elements of the VVC video encoder.

In the context of the above-mentioned works, other proposals were also considered, in which, in addition to a more accurate estimation of probabilities, a more sophisticated context modeling (so, the way of division of stream of binary symbols into independent sub-streams) was considered for selected syntax elements data [34]. Such joint approach results in 1% gain of coding efficiency on average (when testing in the high efficiency compression mode), when compared to the original CABAC technique in HEVC.

A separate, small group of proposals are those that use artificial neural networks [30], [31], [33]. However, these studies should still be considered as preliminary. The proposed methods are only applied to single syntax elements (and not to all syntax elements that are encoded by arithmetic encoder), which in practice makes it impossible to assess the actual performance of the method.

Efforts to improve the CABAC algorithm were also the subject of the author's activities. In the previous years, the author has developed an original solution of improving CABAC through the application of a precise mechanism of probability estimation that uses the Context-Tree Weighting (CTW) method, well-known in the literature [16], [17]. The author has proposed a unique way of adopting CTW in CABAC, which together with the test results was presented in a series of publications [18]–[21]. That solution of the author has been named CTW-CABAC. In the HEVC video encoder these proposals by the author helped reduce the data stream at the CABAC output by 0.7% on average, which made these proposals competitive to other known improvements of CABAC.

C. THE GOAL OF THE PAPER

The precise estimation of probabilities of data symbols for application in entropy encoding is the subject of the paper. The goal is to propose a new, more sophisticated algorithm of probability estimation for use in the CABAC algorithm. The paper presents details of the proposed algorithm, the author's idea of incorporating the algorithm into the CABAC method, and reports the parameters (complexity and compression efficiency) of the proposed CABAC+ technique that employs the author's algorithm of probability estimation.

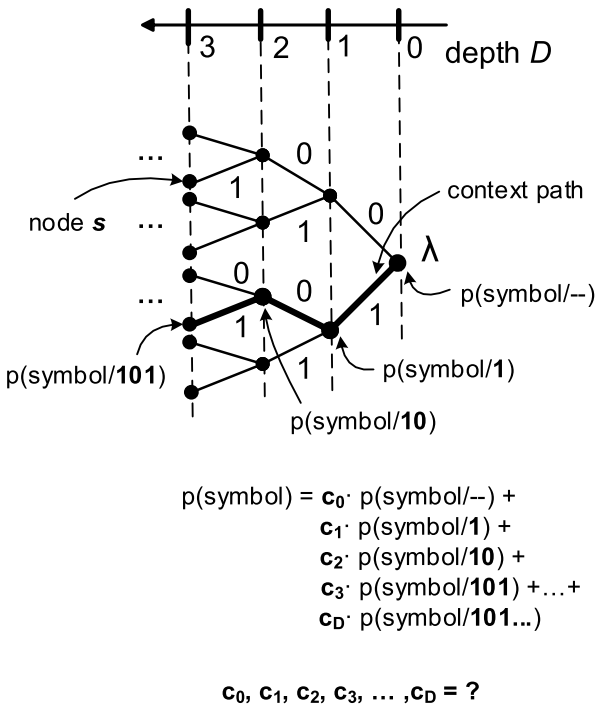

FIGURE 3. The idea of calculating the probability of a symbol as a weighted sum of specific number of different conditional probabilities. The result of this weighted sum is the probability value that is used by the arithmetic encoder.

III. PROPOSED ALGORITHM OF PROBABILITY ESTIMATION

A. MAIN IDEA OF THE SOLUTION

The syntax elements of a video data stream are a very complex type of data. The value of the currently coded syntax element data depends on the values of elements that have been encoded previously. However, it is not known *a priori* what kind of relationship this is. For this reason, good idea is estimating some number of different values of probabilities for a data symbol, instead of just one value of probability. It is said about the values of conditional probabilities that a given data symbol will have a specific value. Because it is impossible to determine (at the side of the decoder) which among the probabilities "predicts" the value of the current data symbol most accurately, it is worth weighting some number of different probabilities. As a result, the final probability is obtained, denoted as $p(symbol)$, that can be used in the entropy encoder. The idea of this approach is illustrated in the figure below (Fig. 3) for the case of probability estimation for binary symbols.

In order to calculate the values of probabilities, one needs to collect information about the number of symbols (and their values) that have been encoded so far. For this purpose, the author uses the binary tree (see Fig. 3) due to processing binary symbols. The binary tree is a collection of nodes s

that are connected by branches. In a given node s of the tree there is information stored about the number a of symbols 0 and the number b of symbols 1 that appeared so far in a situation, when the previously encoded binary symbols had specific values. Those previously encoded symbols are called the context. On the binary tree the context (denoted by ctx_s) is formed by the binary symbols that are assigned to individual branches of the tree, that belong to the path (more precisely called the context path) connecting the root λ of the tree with a given node s. Therefore, it is possible to calculate different conditional probabilities in individual nodes s of the tree, as different contexts ctx_s result in different sequences of previously encoded symbols. This way, each node s represents specific conditional probability $p\,(symbol/ctx_s)$, for example, the conditional probability $p\,(symbol/101)$ of the current binary symbol, when the sequence ctx_s of the three previously encoded symbols was 101. The number of different conditional probabilities $p\,(symbol/ctx_s)$ that can be calculated this way depends on the depth of the binary tree used, which is denoted as D (see Fig. 3).

As already mentioned before, from the point of view of entropy compression of symbols, it is worth using some number of different conditional probabilities values instead of just one value. From the mathematical point of view, the value $p\,(symbol)$ of symbol probability is calculated as a weighted sum of D+1 different conditional probabilities, which were calculated for different contexts ctx_s:

$$p\,(symbol) = \sum_{i=0}^{D} c_i \cdot p\,(symbol/ctx_s) \tag{1}$$

This idea is not new and has already been published in the literature [22] for the purpose of compressing other types of data than video. This paper proposes to use the idea for video data. To perform the weighting of probabilities, c_i ($i = 0, 1, \ldots, D$) weighting coefficients are used, as depicted in Fig. 3 and presented in formula (1). The method of determining the c_i coefficients, as well as the values of individual conditional probabilities, influences the efficiency of the statistics estimation method. The above issues are discussed in the next sections of the paper.

B. METHOD OF DETERMINING CONDITIONAL PROBABILITIES

In this work it is proposed to calculate probabilities $p\,(symbol/ctx_s)$ in nodes s of the tree with the Krichevski-Trofimov (KT) estimator [23]. This estimator is used in data compression, which is confirmed, for example, by its application in the Context-Tree Weighting Method [16], [17].

According to the KT estimator, probability $p_{KT}(0)$ of symbol 0 and probability $p_{KT}(1)$ of symbol 1 are expressed by:

$$p_{KT}\,(0) = \frac{a + 0.5}{a + b + 1}, \quad p_{KT}(1) = \frac{b + 0.5}{a + b + 1} \tag{2}$$

where a and b denotes the number of 0 and 1 symbols, respectively that have been coded so far.

The conditional probabilities that are obtained in nodes s of the context path are then the subject of the weighting procedure, in order to obtain the final probability $p\,(symbol)$ of a symbol (see illustration in Fig. 3 and mathematical formula (1)). For the purpose of this procedure, the weighting coefficients c_i ($i = 0, 1, \ldots, D$) are calculated with the use of the gradient-based optimization algorithm. In practice, the calculation of weighting coefficients c_i ($i = 0, 1, \ldots, D$) is carried out independently for each possible context path on the tree. According to the author's knowledge, this approach is completely new when using it for the purpose of entropy compression of video data. The high efficiency of this solution has already been confirmed in the case of compression of other types of data, such as text data. A good example here are some versions of the PAQ algorithm [22]. An important novelty of this work is showing mathematical formulas for the algorithm (formulas were derived by the author), and additionally, presenting the ways the algorithm can be used in a contemporary entropy encoder, such as the CABAC technique.

C. THE METHOD OF DETERMINING WEIGHTING COEFFICIENTS

The method of calculation of weighting coefficients c_i ($i = 0, 1, \ldots, D$) has a particularly strong impact on the efficiency of the algorithm. The obvious goal is to maximize this efficiency, i.e., to minimize the number of bits that are produced by the entropy encoder. In this work, the aim is to choose such values of weighting coefficients c_i ($i = 0, 1, \ldots, D$) that directly minimize the bit cost of data encoding. As mentioned before, this minimization is realized in the proposed algorithm separately for each possible context path on the binary tree.

The abovementioned idea of the weighting coefficients calculation is successfully used in the PAQ text compression algorithm. Based on this idea, the author has made an appropriate mathematical derivation (see below) to propose formulas that can be used in the video encoder and the video decoder. Thus, the formulas presented below are part of the author's solution that is presented in this paper.

As it is known from information theory, the bit cost of encoding a symbol, e.g., with the value of 0, depends on probability p_0 of the occurrence of this symbol. The same applies to the symbol with the value 1 (whose probability of appearance is $p_1 = 1 - p_0$) – this cost depends on p_1. For the 0 symbol this bit cost is:

$$f = log_2\left(\frac{1}{p_0}\right) \text{ [bits]} \tag{3}$$

In the case of the presented algorithm, probability p_0 of the 0 symbol (observed in a given context ctx_s) is the weighted sum of a given number of conditional probabilities, as it was presented earlier. To be sure that coefficients c_i ($i = 0, 1, \ldots, D$) are fractions (so are lower than 1), and their sum is equal to 1, they are defined by new coefficients w_i ($i = 0, 1, \ldots, D$), when using the following relation:

$$c_i = \frac{w_i}{\sum_i w_i} \tag{4}$$

Using such a notation, the weighted probability p_0 of symbol 0 is defined as:

$$p_0 = \frac{\sum_i w_i \cdot p_{0i}}{\sum_i w_i} \tag{5}$$

where p_{0i} is the conditional probability of symbol 0 calculated in the i-th tree node s of the context path.

We are interested in making the bit cost of encoding a symbol as small as possible. So, in the case of encoding symbol 0, the goal is to minimize the following objective function:

$$f\left(\underline{w}\right) = log_2\left(\frac{1}{p_0}\right) = log_2\left(\frac{\sum\limits_i w_i}{\sum\limits_i w_i \cdot p_{0i}}\right) \rightarrow min \tag{6}$$

where $\underline{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_D \end{bmatrix}$.

So, we ask the question, what the values of w_i ($i = 0, 1, \ldots, D$) should be in order to reach the minimum value for the function $f\left(\underline{w}\right)$? Our task is therefore optimization problem.

Such an optimization can be efficiently carried out using the Cauchy optimization method [24], [25], which belongs to the class of gradient-based optimization algorithms. According to this method, the optimal values of coefficients w_i (i.e., the values for which function $f\left(\underline{w}\right)$ achieves the minimum value) can be determined from the following iterative formula:

$$\underline{w}^{(k+1)} = \underline{w}^{(k)} - \nabla f\left(\underline{w}^{(k)}\right) \tag{7}$$

where:

$\underline{w}^{(k)}$– vector of coefficients w_i obtained in the k-th iteration;

$\underline{w}^{(k+1)}$– vector of coefficients w_i obtained in the (k+1)-st iteration;

$\nabla f\left(\underline{w}^{(k)}\right) = \begin{bmatrix} \frac{\partial f\left(w_0^{(k)}\right)}{\partial w_0} \\ \frac{\partial f\left(w_1^{(k)}\right)}{\partial w_1} \\ \vdots \\ \frac{\partial f\left(w_D^{(k)}\right)}{\partial w_D} \end{bmatrix}$ – gradient of function $f\left(\underline{w}\right)$ in $\underline{w}^{(k)}$, so the vector of the first partial derivatives of f, calculated in $\underline{w}^{(k)}$.

In the case of the i-th element of vector $\underline{w}$ (i.e., the coefficient w_i), the first partial derivative of f is expressed by the formula:

$$\frac{\partial f}{\partial w_i} = \left(log_2\left(\frac{1}{p_0}\right)\right)' = \left(log_2\left(\frac{\sum\limits_i w_i}{\sum\limits_i w_i \cdot p_{0i}}\right)\right)'$$

$$= \frac{1}{ln2 \cdot \frac{\sum\limits_i w_i}{\sum\limits_i w_i \cdot p_{0i}}} \cdot \left(\frac{\sum\limits_i w_i}{\sum\limits_i w_i \cdot p_{0i}}\right)' \tag{8}$$

Performing the appropriate mathematical operations leads to:

$$\frac{\partial f}{\partial w_i} = \frac{1}{ln2} \cdot \left(\frac{1}{\sum\limits_i w_i} - \frac{p_{0i}}{\sum\limits_i w_i \cdot p_{0i}}\right) \tag{9}$$

The i-th element of vector $\underline{w}$ can be calculated using the formula:

$$w_i^{(k+1)} = w_i^{(k)} - \frac{\partial f\left(w_i^{(k)}\right)}{\partial w_i} \tag{10}$$

The mathematical apparatus presented above works for the case of determining the weighted probability of symbol 0. In the case of symbol 1, the idea is exactly the same. However, the final expression on the values of coefficients w_i will be slightly different, because the equation for probability p_1 of symbol 1 is different. This probability is equal to:

$$p_1 = 1 - p_0 \tag{11}$$

So, the bit cost of encoding symbol 1 is:

$$f = log_2\left(\frac{1}{p_1}\right) = log_2\left(\frac{1}{1 - p_0}\right) \text{ [bits]} \tag{12}$$

Function $f\left(\underline{w}\right)$ that is the subject of minimization has the form:

$$f\left(\underline{w}\right) = log_2\left(\frac{1}{1 - p_0}\right) = log_2\left(\frac{1}{1 - \frac{\sum_i w_i \cdot p_{0i}}{\sum_i w_i}}\right)$$

$$= log_2\left(\frac{\sum_i w_i - \sum_i w_i \cdot p_{0i}}{\sum_i w_i}\right) \rightarrow min \tag{13}$$

The first partial derivative of this function, calculated over the i-th element of vector $\underline{w}$ (so, over the variable w_i) is described by the expression:

$$\frac{\partial f}{\partial w_i} = \left(log_2\left(\frac{1}{1 - p_0}\right)\right)' = \left(log_2\left(\frac{1}{1 - \frac{\sum_i w_i \cdot p_{0i}}{\sum_i w_i}}\right)\right)'$$

$$= \frac{1}{ln2} \cdot \left(\frac{1}{\sum_i w_i} - \frac{1 - p_{0i}}{\sum_i w_i - \sum_i w_i \cdot p_{0i}}\right) \tag{14}$$

Individual coefficients w_i are determined using formula (10).

After encoding the binary symbol (0 or 1) elements of vector $\underline{w}$ are updated, when realizing one iteration of formula (10). The first partial derivative $\frac{\partial f\left(w_i^{(k)}\right)}{\partial w_i}$ is calculated using formula (8) or (14), depending on the value of the just encoded (or decoded) binary symbol (0 or 1).

IV. IMPROVEMENT OF CABAC WITH THE PROPOSED ALGORITHM – CABAC+ TECHNIQUE

A. REALIZATION OF THE ALGORITHM IN THE CABAC TECHNIQUE

In order to demonstrate the efficiency of the proposed algorithm of probability estimation in the framework of the advanced entropy encoder, the algorithm has been applied in the CABAC technique. The version of CABAC that is exploited in the HEVC video compression technology has been used as a starting point [32]. Obtained improved CABAC algorithm is called CABAC+ in this paper.

The method of application of the algorithm in the entropy encoder strongly affects the final efficiency of data encoding. Therefore, the author has proposed an original, sophisticated method of incorporating the algorithm presented above into CABAC. Having in mind the advantages of the solutions used in the original CABAC technique, its structure has remained intact, except for the stage of determining the probabilities of symbols (0 and 1). In the original CABAC those probabilities are calculated in a simplified manner, using a high number of 173 predetermined Finite State Machines (FSMs) [8], [9]. In the proposed CABAC+ technique those machines were replaced by the algorithm of symbol probability estimation that is presented in this paper. This way, instead of 173 independent FSMs, 173 separate instances of the proposed algorithm have been introduced in CABAC. In the CABAC+, each such instance of the algorithm estimates the probability of symbols for a specific sub-stream of binary data out of 173 defined in CABAC. For this purpose, the CABAC+ exploits a set of 173 binary trees. Each binary tree gathers information about the statistics of symbols 0 and 1 within individual data sub-streams and for individual context ctx_s (previously encoded symbols). The final probabilities of symbols are calculated based on the information collected in the proper nodes s of the binary trees.

The number of different conditional probabilities that are the subject of weighting in the proposed algorithm may influence the efficiency of the CABAC+ technique. This number depends on depth D of binary trees. Therefore, in this paper, the efficiency of the CABAC+ is explored as a function of D.

The values of probabilities that are calculated by the proposed algorithm cannot be directly input to the arithmetic encoder core of CABAC. It due to the fact that the CABAC core has been designed to work with a limited, predefined set of probabilities [26]. It was done to speed up the CABAC technique. The proposed algorithm of probability estimation produces probabilities that belong to a much wider set of values. Therefore, to meet the abovementioned limitations of the CABAC arithmetic encoder core, the values of probabilities obtained with the proposed algorithm must be mapped to one of the allowed values in original CABAC (it is a set of 127 pre-defined values of probabilities). The criterion of this mapping is to minimize the absolute difference between two probabilities: one calculated with the proposed algorithm of probability estimation, and one allowed by the CABAC arithmetic encoder core.

B. INITIALIZATION OF PARAMETERS OF THE CABAC+

Efficient operation of the CABAC+ technique requires its proper initialization before the compression of video data. The initialization concerns two types of parameters: 1) the numbers a and b of 0 and 1 symbols, determined in each node s of the binary trees of depth D, and 2) weighting coefficients c_i ($i = 0, 1, \ldots, D$) for each possible context path of the binary trees. In the latter case, the coefficients are responsible for the proper weighting of a set of conditional probabilities of binary symbols. The initialization of the abovementioned parameters is performed each time before compressing of an individual video frame.

In order to determine the initial values of these parameters, a separate experiment was carried out, in which selected frames of a few 2K test sequences were combined into one video sequence, and then fully encoded with the HM 16.6 reference software of the HEVC video encoder [27]. The used test sequences and detailed scenario of the encoding followed the methodology highlighted in the next section of the paper. In particular, it was the encoding with the use of I and B frame types, for the values of the quantization parameter (QP) equal to 22, 27, 32, 37. The values of parameters a, b and c_i ($i = 0, 1, \ldots, D$) obtained at the end of such encoding for individual QPs and frame types have been accepted as initial values for real use of the technique.

V. METHODOLOGY OF EXPERIMENTS

The aim of the experiments was to evaluate the performance of the CABAC+ technique. The performance includes both compression efficiency and computational complexity of the technique. The goal was to test the performance of the CABAC+ when encoding syntax elements data that are the subject of entropy compression in the HEVC video coding technology [5], [7]. For this purpose, the CABAC+ technique was implemented and activated in the HM 16.6 reference software [27] of the HEVC video codec [5], [7]. As a result, the modified HEVC codec was obtained. The prepared software was used to carry out a series of tests in which the following data were gathered:

- the quality of encoded video sequence,
- bit cost of representation of a video after compression,
- computational complexity of the proposed technique.

In order to assess the parameters obtained for the proposed solution, the abovementioned data were confronted with the results obtained when using the original HM 16.6 software that works with the original version of the CABAC technique. In other words, the original CABAC technique was the anchor for the results of the solution that is proposed in this paper.

The purpose of the presented research was to further increase the efficiency of video compression, compared to the case when the original CABAC technique is used. In this context, the solution presented in the paper was tested under the scenario when video encoding was oriented towards obtaining very high compression efficiency of data. In order to fulfill this requirement, both tested video encoders (the modified HEVC and the anchor – original HEVC) were configured in

a way that exploits advanced compression tools listed in the ''random access'' scenario of encoding that has been defined in the ''common test conditions'' recommendation [28]. Also the evaluation of parameters of the proposed technique was carried out in accordance with the ''common test conditions''. Such a scenario of research has its strong, practical justification – it finds practical use in both ''video on demand'' and video streaming applications. The first tests were performed by encoding a first set of 2K test sequences. In this part of the experiments, the sequences *BasketballDrive*, *BQTerrace*, *Cactus*, *Kimono1*, *ParkScene* were used. In subsequent tests also 4K test sequences were used together with a second set of 2K sequences. These were 4K sequences: *CampfireParty*, *CatRobot*, *DaylightRoad*, *ToddlerFountain*, *Parkjoy*, *PeopleOnStreet*, *Traffic*, and 2K sequences: *Tennis*, *Riverbed*, *Toys and calendar*, *Walking couple*. All used sequences were found recommendation of the ISO/IEC MPEG and ITU-T VCEG groups of experts as the right testing material for studies on new technologies on video compression. In accordance with recommendation [28], each of the sequences was encoded four times by each of the two encoders (the modified and the original), setting different values of the quantization parameter (QP) in each of the four encodings. The values of QP were as follows: 22, 27, 32, 37. They correspond to encoding a video in which the subjective quality of decoded video frames vary from high (QP = 22) to low (QP = 37). The quality of the decoded video was measured by an objective peak signal-to-noise ratio (PSNR) metric, which is commonly adopted in works on image and video compression. For each of the video sequences, the data of four measuring points (bitrate and PSNR for the values of QP = 22, 27, 32, 37) obtained in the two encoders (the modified and the original) were then compared to each other using a commonly known Bjontegaard metric (denoted as BD-BR) [29]. This metric indicates in percentage terms, what is the saving in bitrate between two bitrate-PSNR curves (obtained for the two compared video encoders), when receiving decoded video frames of the same objective quality. Such a metric was calculated for each video sequence that was used in the tests.

The characteristic feature of the developed technique is the possibility to set the number of the conditional probabilities to be used in the procedure of calculating the final probability of a symbol (in the course of weighting the probabilities). It is obvious that this number can affect the results of the technique. But the question is – to what extent? In order to explore this issue, the proposed technique was tested when using different numbers of conditional probabilities. In practice, it was realized by setting up a different depth D of binary trees in the CABAC+ technique.

In addition to compression efficiency tests, the goal was also to examine the computational complexity of the proposed technique. The commonly used practice is to express the complexity of the encoder and the decoder as the time of program execution by a computer processor. This part of the research was also carried out for different numbers of conditional probabilities weighted in the proposed technique.

TABLE 1. Comparison of coding efficiency of the CABAC+ and the original cabac technique. The table presents BD-BR savings of the CABAC+ over original CABAC. Negative numbers mean compression gain.

	BD-BR [%] (CABAC+ vs original CABAC)			
2K video sequence	D=2	D=4	D=6	D=8
BasketballDrive	-0.50%	-0.64%	-0.63%	-0.58%
BQTerrace	-0.65%	-0.82%	-0.83%	-0.77%
Cactus	-0.44%	-0.60%	-0.63%	-0.61%
Kimono1	-0.67%	-0.79%	-0.80%	-0.77%
ParkScene	-0.71%	-0.87%	-0.94%	-0.96%
Average:	**-0.59%**	**-0.74%**	**-0.77%**	**-0.74%**

On the basis of partial results, an attempt was made to select the optimal settings of the proposed technique that ensure the best compromise between its coding efficiency and computational complexity.

VI. RESULTS OF EXPERIMENTS

A. EFFICIENCY OF THE CABAC+ TECHNIQUE

1) RESULTS FOR THE FIRST VERSION OF THE TECHNIQUE

This section presents the results for the version of the technique in which the Cauchy optimization algorithm was initialized (at the beginning of each video frame) with default values of weights w_i ($i = 0, 1, \ldots, D$). In this version, the solution proposed in this paper is characterized by noticeably higher coding efficiency than the original CABAC technique. When operating in the framework of the HEVC video encoder the bitrate savings (so increase in coding efficiency) ranges from 0.5% to 0.8%, depending on the adopted setting of the proposed technique (see results of Table 1). This setting is the number ($D + 1$) of different probabilities that are the subject of the weighted sum in the proposed algorithm.

When using a small value of $D = 2$ the observed gain of coding efficiency was 0.44% to 0.71%, depending on the content of the test sequence. It was 0.59% on average (see Table 1 for more details). However, the additional increase of D by 2 (so, to the value $D = 4$) brought a further increase of coding gain to the average value of 0.74%. As before, the exact result of bitrate saving was depended on the test sequence, ranging from 0.6% to 0.87%.

In this phase of experiments the best results were obtained for parameter $D = 6$. However, the improvement of the coding efficiency of data compared to $D = 4$ is small. The average gain of efficiency (in relation to the original CABAC) is 0.77%, which is a better result only by 0.03% when compared to the average result for $D = 4$. The depth of trees $D = 6$ for the proposed technique is the value for which the results are just saturated. Further increase of the value of D does not bring any additional improvement in efficiency, and even leads to slight deterioration. It is confirmed by the results for $D = 8$. From the point of view of the presented results, it is apparent that $D = 4$ is the setting that allows us to reach the best compromise between coding efficiency and complexity of the technique.

2) RESULTS FOR THE UPDATED VERSION OF THE TECHNIQUE

From the practice of optimization it is commonly known that that the starting point of the Cauchy algorithm may strongly affect the final results. In order to take this aspect into account, the updated version of the proposed technique has also been considered in the research. In this version, the Cauchy optimization method has been using dedicated starting values of weights w_i ($i = 0, 1, \ldots, D$). Those dedicated weights were determined as the final values that were obtained during the operation of the method for fragments of several test sequences.

This part of experiments was carried out for the highest tested value of $D = 8$, for both 2K and 4K spatial resolution test sequences. In the 2k sequence set there were sequences from the first and second sets (see section V for detailed list of test sequences). For comparison purposes, experiments were performed for cases when the default or dedicated values of weights w_i ($i = 0, 1, \ldots, D$) were used for individual context paths of binary trees, during the encoding of consecutive frames of a video. The obtained results are presented in Table 2.

The main conclusion derived from the research is that indeed, the ''improvement'' of the starting point of the Cauchy method made it possible to get even better results of bitrate savings – on average by nearly 0.1%. Such a result was obtained for the set of 2K sequences. This may seem very small, but when taking into account that the entire profit of efficiency is less than 1%, such an additional increase of encoding efficiency should already be considered as noticeable. More so, because it relates to a lossless coding technique.

An interesting observation can be made in the case of results for 4K test sequences. Here, the additional gains of encoding efficiency are smaller when using dedicated values of weights w_i ($i = 0, 1, \ldots, D$), instead of their default values. On average, this gain is 0.03%. In the author's opinion, this regularity can be explained by the fact that a single frame of a 4K video contains much more data than a frame of 2K sequence. In the case of a larger amount of frame data, the algorithm has a better possibility to determine more proper values of w_i weights (during data encoding), even when the encoding starts with the default values.

3) DETAILED RESULTS OF EFFICIENCY OF THE CABAC+ TECHNIQUE

The Bjontegaard metric (BD-BR) values presented in the previous section were determined based on the partial results of video encoding, carried out for 4 values of the QP parameter: QP = 22, 27, 32, 37 (as highlighted in Section V). There is where we can address the question about efficiency of the CABAC+ technique for individual QP values. To answer this question the 'Rate – PSNR' curves are presented here (see Fig. 5, 6, and 7), which have been generated for the selected test video sequences. The marked points of the 'Rate – PSNR' curves correspond to the results of video encoding obtained for the original CABAC and CABAC+ at a given value of the QP parameter. The presented results relate to the variant of the updated version of the CABAC+ technique and depth D of context trees equal to 8.

As can be seen in the charts, the CABAC+ outperforms the original CABAC in the entire considered range of bitrates. The dashed curve that represents the results for CABAC+ is always located above the graph with the results for the original CABAC, which means that CABAC+ technique allows us to obtain higher quality of the encoded video at the same bitrate. What is worth noting, the higher the bitrate, the slightly bigger the difference in the efficiency of tested encoders. This is especially seen in the graph for the *Traffic*

TABLE 2. Comparison of coding efficiency of the CABAC+ and the original CABAC technique. The table presents BD-BR savings of the CABAC+ over original CABAC. Negative numbers mean compression gain.

	BD-BR [%] (CABAC+ vs original CABAC)	
2K video sequence	**D=8#**	**D=8***
BasketballDrive	-0.58%	-0.67%
BQTerrace	-0.77%	-0.84%
Cactus	-0.61%	-0.73%
Kimono1	-0.77%	-0.84%
ParkScene	-0.96%	-1.07%
Tennis	-0.74%	-0.75%
Riverbed	-0.50%	-0.54%
Toys and calendar	-0.73%	-0.85%
Walking couple	-0.89%	-0.96%
Average:	**-0.73%**	**-0.81%**
4K video sequence	**D=8#**	**D=8***
CampfireParty	-0.73%	-0.79%
CatRobot	-0.94%	-1.00%
DaylightRoad	-0.80%	-0.85%
ToddlerFountain	-1.03%	-1.04%
Parkjoy	-1.19%	-1.20%
PeopleOnStreet	-1.12%	-1.14%
Traffic	-2.58%	-2.62%
Average:	**-1.20%**	**-1.23%**
# first version of the technique (use of default starting values of weights w_i)		
* updated version of the technique (use of dedicated starting values of weights w_i)		

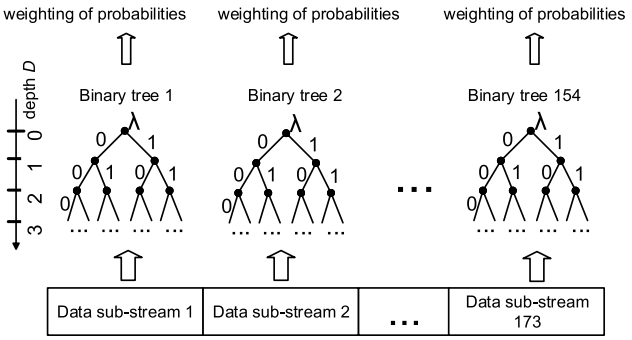

FIGURE 4. The method of adopting the presented algorithm of probability estimation in the CABAC technique.

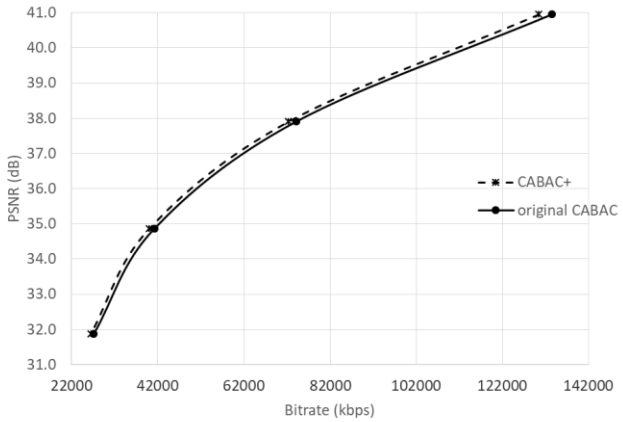

FIGURE 5. 'Rate – PSNR' curves showing coding efficiency of the two encoders: (1) CABAC+ and (2) original CABAC. Results for Traffic test video sequence. Results for the HM 16.6 software.

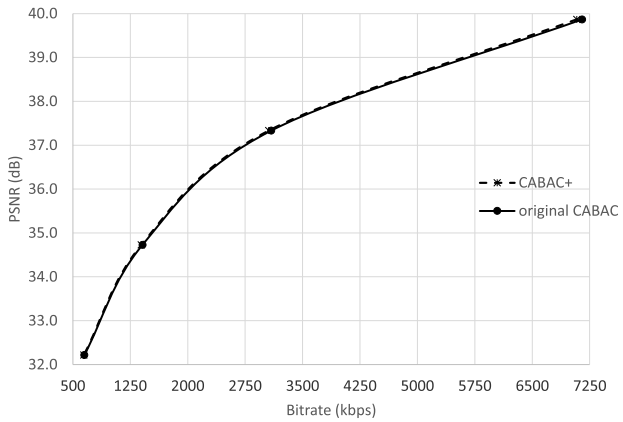

FIGURE 6. 'Rate – PSNR' curves showing coding efficiency of the two encoders: (1) CABAC+ and (2) original CABAC. Results for ParkScene test video sequence. Results for the HM 16.6 software.

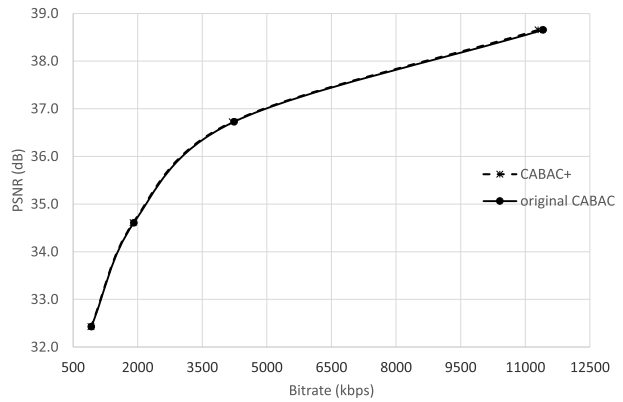

FIGURE 7. 'Rate – PSNR' curves showing coding efficiency of the two encoders: (1) CABAC+ and (2) original CABAC. Results for *Walking couple* test video sequence. Results for the HM 16.6 software.

sequence. The entropy encoder encodes higher amounts of data for higher bitrates. It allows the proposed technique to estimate the probabilities of symbols more accurately than it is the case for lower bitrates.

TABLE 3. Comparison of complexity of the two HEVC video decoders: the modified that works with the CABAC+, and the original using the original CABAC. Relative results obtained by referencing to the runtime of the original HEVC decoder.

	relative decoding time of a video: T_{CABAC+} / $T_{original\ CABAC}$				
2K video sequence		**D=2**	**D=4**	**D=6**	**D=8**
BasketballDrive	QP=22	1.39	1.43	1.82	1.93
	QP=27	1.17	1.20	1.31	1.44
	QP=32	1.10	1.17	1.21	1.25
	QP=37	1.08	1.08	1.11	1.14
BQTerrace	QP=22	1.41	1.54	1.85	2.05
	QP=27	1.05	1.06	1.17	1.23
	QP=32	0.91	0.91	0.93	0.98
	QP=37	0.89	0.90	0.89	0.95
Cactus	QP=22	1.29	1.36	1.69	1.81
	QP=27	0.91	0.98	1.04	1.14
	QP=32	1.07	1.08	1.20	1.22
	QP=37	1.05	1.02	1.10	1.12
Kimono1	QP=22	0.94	1.00	1.13	1.20
	QP=27	1.00	1.04	1.14	1.19
	QP=32	0.96	0.99	1.04	1.08
	QP=37	0.94	0.97	1.02	1.03
ParkScene	QP=22	1.28	1.23	1.41	1.53
	QP=27	0.99	1.05	1.16	1.23
	QP=32	0.96	0.99	1.07	1.10
	QP=37	0.79	0.79	0.81	0.84
	Average:	**1.06**	**1.09**	**1.20**	**1.27**

B. COMPLEXITY OF THE CABAC+ TECHNIQUE

The higher efficiency of the proposed solution is obtained at the cost of higher complexity of the technique. It refers to both computational complexity and memory usage.

As experiments have revealed, higher complexity of the entropy encoder does not translate in practice into longer operation time T of the video encoder. In the hybrid video encoder there are other mechanisms, like motion estimation and coding mode selection for image blocks, which are much more computationally complex than the entropy coding of data. However, the situation is different on side of the video decoder. Here, the contribution of the entropy decoding part to the total decoding time of a video is much higher.

Thus, in the HEVC video decoder, the proposed entropy coding technique extends video decoding time T by 6% - 27% on average, depending on the used depth D of binary trees. Of course, the higher percentage increase of complexity of the decoder is related to the greater depth D of trees, while this increase is smaller for smaller values of D. Additionally, the increase in decoding complexity is strongly dependent on the value of QP and, to a lesser extent, on the content of test sequence, which is well illustrated by the numerical values of Table 3. A smaller QP translates into a larger amount of decoded data (a higher number of binary symbols which are decoded by the entropy decoder). Increasing the operating time of the entropy decoder is therefore more visible in this case. To some extent the content of a video sequence also affects the amount of data being entropy-encoded, thus the variation of results for different test sequences (see Table 3).

The proposed algorithm also presents a greater demand for memory usage. First of all, the counters a and b of the number of 0 and 1 symbols must be stored in each node s

of the binary trees used (in the case of implementation in the HM 16.6 software, 173 different binary trees are used). In addition to this, the next container of data must also be associated with each of the above-mentioned binary trees. This container stores the values of weighting coefficients w_i ($i = 0, 1, \ldots, D$) for each possible context path of a binary tree, and is necessary for handling the Cauchy optimization algorithm. Having regard to the number of different binary trees used (173 trees), the number of nodes s in a single binary tree (e.g., 127 nodes in a tree for $D = 6$), the number of different context paths on a tree (64 different paths for a tree of depth $D = 6$), the number of nodes on a single context path (it is $D+1$), and assuming that each of the single coefficients (a, b and w_i) is represented by one byte, it gives in total about 40 kbytes of data to save a and b counters, and nearly 70 kbytes to store the weighting coefficients w_i. This memory usage is distinctly higher than in the original CABAC (in the original CABAC it is less than 1 kbyte), but the quoted values still seem to be relatively low, mainly from the point of view of the software implementation of the technique and capabilities of current processors. This memory consumption increases, of course, with the increase of D, but, as research has shown, depth $D = 6$ is the best setting from the point of view of technique efficiency.

C. COMPARISON TO OTHER IMPROVED VERSIONS OF CABAC

The previous sections have confronted the parameters of the proposed technique with the original CABAC. The question is about the efficiency and complexity of the proposed technique when compared to other, more advanced methods than CABAC that were developed recently. This subsection concerns this issue.

To realize this goal, the technique proposed in this paper has been confronted with the author's other proposal of the improved CABAC, which has been presented in many publications of the author [18]–[21]. We are talking about the improved CABAC technique, called CTW-CABAC (refer to section II B). The choice of CTW-CABAC as a reference technique in these studies was dictated by the following premises. First, CTW-CABAC exceeds the original CABAC method in terms of efficiency of video data encoding. It has also been shown in many studies as a competitive solution to the latest available methods of entropy coding [18]. Second, the author has a full, own program implementation of the CTW-CABAC technique (also in the HM 16.6 software), which allows for a precise comparison of both algorithms (CTW-CABAC and the CABAC+ technique presented in this paper) within the same scenario of video compression. The adopted scenario of studies is as described earlier in the methodology of experiments.

1) COMPARISON OF CODING EFFICIENCY

The obtained results clearly show that in the vast majority of cases the proposed CABAC+ technique outperforms CTW-CABAC in its data encoding efficiency (see values in Table 4). This applies especially to small values of binary tree depth D. In the results, it is clearly visible that the lower the value of D, the higher the efficiency improvement in relation to CTW-CABAC. As one can see for 2K sequences, for $D = 2$, this improvement is 0.19% on average, while for $D = 4$, the increase of coding efficiency is 0.12% on average. Even greater improvements were obtained for 4K sequences – it was 0.26%, 0.18%, and 0.10% respectively. In the conducted research, the efficiency of CTW-CABAC was higher than the efficiency of the proposed technique only for $D = 8$ (except for the two 4K sequences). It was rather small difference of 0.05% – 0.1% on average. The presented results confirm what could previously have been suppressed, that in the case of weighting a small number of conditional probabilities (the case of small value of D), proper selection of weighting factors is of particular importance. And in fact, the proposed technique uses dedicated values of weighting factors (calculated with the Cauchy optimization method), which cannot be said about the CTW-CABAC technique.

TABLE 4. Comparison of coding efficiency of the CABAC+ presented in this paper and the CTW-CABAC technique. Results for syntax elements compression in the HEVC encoder, for different value of D. The table presents BD-BR savings of the CABAC+ over CTW-CABAC. Negative numbers mean compression gain.

	BD-BR [%] (CABAC+ vs CTW-CABAC)			
2K video sequence	**D=2**	**D=4**	**D=6**	**D=8**
BasketballDrive	-0.21%	-0.12%	0.00%	0.14%
BQTerrace	-0.17%	-0.14%	-0.04%	0.08%
Cactus	-0.23%	-0.14%	-0.03%	0.11%
Kimono1	-0.14%	-0.08%	0.01%	0.11%
ParkScene	-0.20%	-0.14%	-0.07%	0.05%
Average:	**-0.19%**	**-0.12%**	**-0.02%**	**0.10%**
4K video sequence	**D=2**	**D=4**	**D=6**	**D=8**
CampfireParty	-0.30%	-0.14%	0.00%	0.26%
CatRobot	-0.28%	-0.15%	-0.06%	0.05%
DaylightRoad	-0.22%	-0.14%	-0.05%	0.05%
ToddlerFountain	-0.16%	-0.14%	-0.08%	0.01%
Parkjoy	-0.22%	-0.18%	-0.11%	-0.03%
PeopleOnStreet	-0.18%	-0.14%	-0.10%	-0.04%
Traffic	-0.44%	-0.35%	-0.29%	0.08%
Average:	**-0.26%**	**-0.18%**	**-0.10%**	**0.05%**

2) COMPARISON OF COMPLEXITY

As the results revealed, the relation between the complexities of the compared techniques depends on the used depth D of binary trees. Generally, the higher value of D, the greater the difference.

And so, in the case of a small value of $D = 2$, the total decoding time T of a video with the modified HEVC decoder (that works with the proposed CABAC+ technique) is the same as for the modified HEVC decoder with CTW-CABAC. This is shown in the results given in Table 5. However, as the value of D increases, the complexity difference increases in favor of the video decoder with the CTW-CABAC technique. However, these are not very significant differences, and for

TABLE 5. Comparison of complexity of the two HEVC video decoders: the first working with the CABAC+ (presented in this paper), and the second operating with the CTW-CABAC technique. Relative results obtained by referencing to the runtime of the HEVC decoder with the CTW-CABAC.

	relative decoding time of a video: T_{CABAC+} / $T_{CTW-CABAC}$				
2K video sequence		**D=2**	**D=4**	**D=6**	**D=8**
BasketballDrive	QP=22	1.09	1.12	1.30	1.24
	QP=27	1.00	1.10	1.12	1.08
	QP=32	0.87	1.13	1.10	1.06
	QP=37	0.91	1.11	1.05	1.01
BQTerrace	QP=22	1.16	1.23	1.31	1.25
	QP=27	0.94	1.10	1.12	1.07
	QP=32	0.91	1.06	1.03	1.03
	QP=37	0.99	1.06	1.00	1.03
Cactus	QP=22	1.11	1.08	1.31	1.22
	QP=27	0.99	1.06	1.14	1.12
	QP=32	0.87	1.01	1.12	1.06
	QP=37	0.90	1.00	1.06	1.02
Kimono1	QP=22	1.06	0.95	1.12	1.14
	QP=27	1.00	1.01	1.11	1.10
	QP=32	1.00	0.98	0.95	1.06
	QP=37	0.91	0.91	1.05	1.05
ParkScene	QP=22	1.23	1.00	1.16	1.17
	QP=27	1.04	1.03	1.08	1.13
	QP=32	1.01	0.98	1.03	1.09
	QP=37	0.97	0.91	1.01	1.05
	Average:	**1.00**	**1.04**	**1.11**	**1.10**

$D = 6$ or $D = 8$ they reach 10% on average. It must be emphasized here that for smaller values of D, for which the efficiency of the proposed algorithm exceeds that for the CTW-CABAC technique (for depth $D < 6$), the complexity of the HEVC decoder with the proposed solution is only slightly higher than the complexity of the video decoder with CTW-CABAC.

VII. CONCLUSION AND FINAL REMARKS

Contemporary algorithms of entropy compression, like the CABAC technique, show really great ability to reduce the statistical redundancy of syntax elements data in the video encoder [1]. Due to the complicated, highly non-stationary nature of syntax elements data, the achievement of the intended goal of high compression efficiency requires the use of complex mechanisms of symbol probability estimation. In the CABAC technique, such complex mechanisms are certainly used.

But this paper shows that the efficiency of the CABAC technique can still be further increased. This however, requires the use of an even more sophisticated algorithm of estimating the probabilities of symbols. In the paper, the author has investigated a new algorithm for estimating probabilities of binary symbols in CABAC. In this new algorithm the values of probabilities are calculated much more accurately than in the original CABAC technique. Instead of determining only one conditional probability value for a binary symbol (as it is the case in the original CABAC) the new algorithm calculates a number of different conditional probabilities of a symbol. It is not known, however, which of these probabilities would be the best to use. Therefore, those probabilities are then appropriately weighted together (with the use of specific weighting factors), in order to obtain the final value of probability that is later used by the arithmetic encoder. In order to determine the proper values of weighting factors, the author has proposed to use the Cauchy optimization method. In the new algorithm, the Cauchy optimization provides such a choice of values of weighting factors that minimizes the number of bits produced by the entropy encoder. Therefore, the probabilities obtained by the proposed new algorithm are, from the point of view of data compression efficiency, better than those from the original CABAC algorithm. The author has also proposed a unique method of incorporating the new algorithm into the CABAC technique, which led to the author's proposal of the CABAC+ entropy encoder. The CABAC+ technique is the essential achievement of the paper.

An important element of the paper are detailed results of experiments that report the coding efficiency and computational complexity of the CABAC+ technique that uses the algorithm of probability estimation presented in the paper. They are the results obtained for a whole video compression system (the video encoder and the video decoder), a wide collection of test video sequences, and different settings of the video encoder. The subject of the research was the MPEG-H HEVC/H.265 video encoder [5], [7]. Currently, works are finalized towards the development of the next generations of video encoding, e.g., the technology of Versatile Video Coding [10]. However, a good research methodology is to use the most advanced existing technology for which there is a version of well-developed software of the video encoder and the video decoder. As of today, this requirement is met by the MPEG-H HEVC/H.265 video compression technology, with its HM reference software [5], [7], [27]. Therefore, this software has been used in this paper as the test platform and reference for the experimental results. The obtained results clearly show that in each single experiment, the proposed CABAC+ entropy encoder proved to be better than the original CABAC, allowing an additional increase of coding efficiency of video data by 0.59% to 1.23%, on average. Detailed results depended on many issues, such as the content of a test sequence, and depth D of binary trees, which was clearly reported and discussed in the paper. What is very important, the CABAC+ technique proposed in the paper also proved to be competitive to a more advanced technique than the original CABAC, as demonstrated by the results of comparing the performance of the proposed technique with CTW-CABAC (see previous section).

The increased efficiency of the CABAC+ technique comes at a price. In the case of the proposed solution, the complexity of entropy encoding and entropy decoding of data is higher. However, this does not apply to the whole video encoder. In the case of the whole video decoder, its operation time was increased by 6% to 20%, when using reasonable, for the proposed technique, values of depth D of binary trees (value of D from 2 to 6). This complexity is slightly larger than for the CTW-CABAC technique [18], [19], by 4% and 11 %, for depths D equal to 4 and 6, respectively. However, these results relate to such a variant of the CABAC+ technique

implementation, in which there are still further possibilities for program code optimization, aimed at further speeding up the calculations.

REFERENCES

[1] D. Karwowski, "Significance of entropy coding in contemporary hybrid video encoders," in *Image Processing and Communications Challenges 4* (Advances in Intelligent Systems and Computing), vol. 184. Berlin, Germany: Springer-Verlag, 2013, pp. 111–117.

[2] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[3] I. H. Witten, J. G. Cleary, and R. Neal, "Arithmetic coding for data compression," *Commun. ACM.*, no. 6, pp. 520–540, Jun. 1987.

[4] A. Said, "Introduction to arithmetic coding—Theory and practice," in *Imaging Systems Laboratory*. Palo Alto, CA, USA: HP Laboratories, Apr. 2004.

[5] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[6] *Generic Coding of Audio-Visual Objects, Part 10: Advanced Video Coding*, Standard ISO/IEC 14496-10, Mar. 2006.

[7] *ISO/IEC and ITU-T, High Efficiency Video Coding (HEVC)*, Standard ISO/IEC 23008-2 (MPEG-H Part 2)/ITU-T Rec. H.265, Apr. 2013.

[8] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 620–636, Jul. 2003.

[9] V. Sze and M. Budagavi, "High throughput CABAC entropy coding in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1778–1791, Dec. 2012.

[10] A. Segall, V. Baroncini, J. Boyce, J. Chen, T. Suzuki, *Joint Call for Proposal on Video Compression With Capability Beyond HEVC*, document ISO/IEC JTC 1/SC 29/WG 11, JVET-H1002, Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3, 8th Meeting, Macao, China, Oct. 2017.

[11] J. Chen, M. Karczewicz, Y.-W. Huang, K. Choi, J.-R. Ohm, and G. J. Sullivan, "The joint exploration model (JEM) for video compression with capability beyond HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, to be published, doi: 10.1109/TCSVT.2019.2945830.

[12] A. Alshin, E. Alshina, and J. Park, "High precision probability estimation for CABAC," in *Proc. Vis. Commun. Image Process. (VCIP)*, 2013, pp. 1–6.

[13] A. Alshin, E. Alshina, and J. Park, *Multi-Parameter Probability Up-Date for CABAC*, document JCTVC-0764, Joint Collaborative Team Video Coding (JCT-VC), Nov. 2011.

[14] J. Stegemann, H. Kirchhoffer, D. Marpe, and T. Wiegand, *Counter-Based Probability Model Update With Adapted Arithmetic Coding Engine*, document JCTVC-547, Joint Collaborative Team Video Coding (JCT-VC), Nov. 2011.

[15] H. Kirchoffer, J. Stegemann, D. Marpe, H. Schwarz, and T. Wiegand, *CE5-Related: State-Based Probability Estimator*, document JVET-K0430, Joint Video Experts Team (JVET) ITU-T SG 16 WP 3 ISO/IEC JTC 1/SC 29/WG, vol. 11, 11th Meeting, Ljubljana, Slovenia, Jul. 2018.

[16] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens, "The context-tree weighting method: Basic properties," *IEEE Trans. Inf. Theory*, vol. 41, no. 3, pp. 653–664, May 1995.

[17] F. M. J. Willems, "The context-tree weighting method: Extensions," *IEEE Trans. Inf. Theory*, vol. 44, no. 2, pp. 792–798, Mar. 1998.

[18] D. Karwowski and M. Domański, "Context-adaptive binary arithmetic coding with precise probability estimation and complexity scalability for high-efficiency video coding," *J. Electron. Imag.*, vol. 25, no. 1, Jan. 2016, Art. no. 013010, doi: 10.1117/1.JEI.25.1.013010.

[19] D. Karwowski and M. Domański, "Increased compression efficiency of AVC and HEVC CABAC by precise statistics estimation," *Int. J. Electron. Telecommun.*, vol. 64, no. 3, pp. 277–284, 2018.

[20] D. Karwowski and M. Domański, "Improved context-adaptive arithmetic coding in H.264/AVC," in *Proc. 17th Eur. Signal Process. Conf. (EUSIPCO)*, Glasgow, Scotland, Aug. 2009, pp. 2216–2220.

[21] D. Karwowski and M. Domański, "Improved arithmetic coding in H.264/AVC using context-tree weighting method," in *Proc. Picture Coding Symp. (PCS)*, Lisboa, Portugal, Nov. 2007.

[22] M. Mahoney, "Adaptive weighing of context models for lossless data compression," Dept. Comput. Sci., Florida Inst. Technol., Tech. Rep. CS-2005-16, 2005. [Online]. Available: http://mattmahoney.net/dc/cs200516.pdf

[23] R. E. Krichevsky and V. K. Trofimov, "The Performance of universal encoding," *IEEE Trans. Inf. Theory*, vol. IT-27, no. 2, pp. 199–207, Mar. 1981.

[24] A. Ravindran, K. M. Ragsdell, and G. V. Reklaitis, *Engineering Optimization: Methods and Applications*. Hoboken, NJ, USA: Wiley, 2006.

[25] S. S. Rao, *Engineering Optimization. Theory and Practice*. Hoboken, NJ, USA: Wiley, 2009.

[26] D. Marpe, G. Marten, and H. L. Cycon, "A fast renormalization technique for H.264/MPEG4-AVC arithmetic coding," in *Proc. 51st Internationales Wissenschaftliches Kolloquium Technische Universitat Ilmenau*, Sep. 2006.

[27] *HEVC Test Model HM Software*. Accessed: Dec. 15, 2017. [Online]. Available: https://hevc.hhi.fraunhofer.de/

[28] F. Bossen, *Common Test Conditions and Software Reference Configurations*, document JCTVC-J1100, Joint Collaborative Team on Video Coding (JCT-VC) of ITUT SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Stockholm, Sweden, Jul. 2012.

[29] G. Bjøntegaard, *Calculation of Average PSNR Differences Between RD Curves*, document VCEG-M33, ITU-T SG16/Q6, 13th VCEG Meeting, Austin, TX, USA, Apr. 2001.

[30] R. Song, D. Liu, H. Li, and F. Wu, "Neural network-based arithmetic coding of intra prediction modes in HEVC," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, St Petersburg, Russia, Dec. 2017, pp. 1–4.

[31] C. Ma, D. Liu, X. Peng, and F. Wu, "Convolutional neural network-based arithmetic coding of DC coefficients for HEVC intra coding," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Athens, Greece, Oct. 2018, pp. 1772–1776, doi: 10.1109/ICIP.2018.8451166.

[32] J. Sole, R. Joshi, N. Nguyen, T. Ji, M. Karczewicz, G. Clare, F. Henry, and A. Duenas, "Transform coefficient coding in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1765–1777, Dec. 2012.

[33] C. Ma, D. Liu, X. Peng, Z. Zha, and F. Wu, "Neural network-based arithmetic coding for inter prediction information in HEVC," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Sapporo, Japan, May 2019, pp. 1–5, doi: 10.1109/ISCAS.2019.8702499.

[34] H. Zhang, L. Song, X. Yang, and Z. Luo, "Evaluation of beyond-HEVC entropy coding methods for DCT transform coefficients," in *Proc. Vis. Commun. Image Process. (VCIP)*, Chengdu, China, 2016, pp. 1–4, doi: 10.1109/VCIP.2016.7805533.

[35] F. L. L. Ramos, A. V. P. Saggiorato, B. Zatt, M. Porto, and S. Bampi, "Residual syntax elements analysis and design targeting high-throughput HEVC CABAC," *IEEE Trans. Circuits Syst. I, Reg. Papers*, to be published, doi: 10.1109/TCSI.2019.2932891.

DAMIAN KARWOWSKI received the M.Sc. and Ph.D. degrees from the Poznan University of Technology, in 2003 and 2008, respectively. He is currently an Assistant Professor with the Poznan University of Technology. He has been taking part in many industry-oriented projects that encompasses video and audio compression. He is the author or coauthor of more than 50 articles on digital video coding in both national and international conferences and journals. His research interests are centered on image and audio compression algorithms, and realization of video and audio codecs on PC and DSP platforms. He is a member of the Faculty of Computing and Telecommunications and the organizing committee of the International Workshop on Signals, Systems, and Image Processing, IWSSIP 2004, the International Conference on Signals and Electronic Systems, ICSES 2004, 73rd Meeting of MPEG, European Signal Processing Conference, and EUSIPCO 2007. He was the Technical Program Chair of the International Workshop on Signals, Systems, and Image Processing, IWSSIP 2017, and the Picture Coding Symposium, PCS 2012.