An MCMC based Efficient Parameter Selection Model for x265 Encoder

Yan Huang*, Li Song*†, Rong Xie*†, Zhengyi Luo‡ and Xiangwen Wang‡
* Shanghai Institute for Advanced Communication and Data Science,
Department of Electronic Engineering, Shanghai Jiao Tong University
† Cooperative Medianet Innovation Center, Shanghai, China
‡ School of Electronics and Information Engineering, Shanghai University of Electric Power
Email: yhuanghy@163.com, song_li@sjtu.edu.cn, xierong@sjtu.edu.cn, lzy@shiep.edu.cn, wxw21st@gmail.com

***Abstract*—As an open-source and computationally efficient High Efficiency Video Coding (HEVC) encoder, x265 has been gaining increasing popularity in video applications. x265 provides numerous encoding parameters in view of flexibility. However, proper and efficient setting of parameters often becomes a great challenge in practice. In this paper, we deeply investigate the influence of x265 parameters based on the Slow preset and pick out important parameters in terms of efficiency and complexity. Then a Markov Chain Monte Carlo (MCMC) based algorithm is proposed for efficient parameter adaptation at the target encoding time. This paper shows that carefully selected low-complexity encoding configurations can achieve the coding efficiency comparable to that of high-complexity ones. Specifically, average 26.72% encoding time reduction can be achieved while maintaining similar Rate Distortion (RD) performance to x265 presets using the proposed algorithm.**

I. Introduction

To improve compression performance relative to existing standards, new generation of video coding standards, including High Efficiency Video Coding (HEVC) [1] and Audio Video Coding Standard 2 (AVS2) [2], have been developed. HEVC has achieved about 40% bitrate reduction than its predecessor H.264/AVC at the same quality. New technologies, such as coding tree block structure and advanced motion vector prediction, have brought about great improvement on compression ratio, but they also lead to significant complexity increasement. Hence, although as the officially provided reference software of HEVC standard, HM [3] can be hardly applied to practical work directly due to its excessive high complexity.

In order to speed up HEVC encoding, many studies, such as parallelism and early termination, have been adopted. HEVC standard adopts Wavefront Parallel Processing (WPP) [4] to facilitate parallel coding. In [5], an efficient prediction scheme is proposed to lower Rate Distortion Optimization (RDO) complexity. A fast Coding Unit (CU) depth decision method is proposed in [6] using the information from adjacent and co-located CUs. Besides local algorithm optimization, practicality of HM can also be improved in a way of complexity control. In order to better employ HM, different complexity-scalable systems have been established in [7], [8], [9], where all parameters work together and important factors are analysed. The results of the analysis are used to control encoding time. Nonetheless, HM is not meant to be a particularly efficient implementation. Complexity control based on HM has only limited practical value.

x265 [10] is the most popular HEVC video encoder nowadays which is meticulously designed so that it's much faster than HM owing to parallelism and some early termination algorithms. Note that since x265 adopts parallelism, the influence on Rate-Distortion-Time (RDT) performance of some parameters shows much difference from single-threaded ones as analysed in [7], [8], [9]. Moreover, x265 has much more parameters than HM due to parallelism and early termination algorithms, so an thorough analysis for the influence of these parameters is highly desired for practical guidance. Unfortunately, to the best of our knowledge, no comprehensive analysis on the influence of x265 parameters has been reported. Thus, the encoding efficiency and complexity can only be adapted by tuning parameters empirically, which often makes parameter setting a great challenge in practice.

In this paper, we firstly give a detailed analysis on RDT performance of x265 parameters and finely choose parameters which are closely correlated to Rate Distortion (RD) performance and encoding time. Then a Markov Chain Monte Carlo (MCMC) [11] based algorithm is proposed to efficiently determine the parameters for target encoding time. It's shown that based on the proposed algorithm, carefully selected low-complexity encoding configurations can achieve coding efficiency comparable to that of x265 presets.

This paper is organized as follows. Section II gives a brief evaluation of x265 encoder. Section III distinguishes important parameters and introduces an MCMC based algorithm for efficient parameter adaptation. Experiments are carried out in Section IV and Section V concludes the paper.

II. x265 Encoder Analysis

In order to apply x265 in an efficient and complexity scalable way, it's necessary to explore its features carefully. To this end, all the default presets of x265 have to be evaluated to obtain RDT related parameters. The analysis is based on x265 (2.3) with Group of Picture (GOP) size 30. The simulations are done in a workstation with Intel Xeon E5-4627 v3, 2.6 GHz, 20 cores and 128 GB RAM Memory. All sequences from Class A to F sequences are tested for each preset and each QP (22, 27, 32 and 37).

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

TABLE I
AVERAGE RDT PERFORMANCE OF X265 DEFAULT PRESETS

Preset	BDBR(%)	BDPSNR(dB)	Time(Normalized)
Placebo	0.0000	0.0000	1.0000
Veryslow	3.2893	−0.2184	0.4462
Slower	4.9443	−0.2820	0.3119
Slow	4.1433	−0.2794	0.1063
Medium	12.9389	−0.6364	0.0483
Fast	17.0600	−0.7601	0.0416
Faster	20.9898	−0.8871	0.0372
Veryfast	21.2165	−0.8965	0.0370
Superfast	31.6217	−1.1786	0.0228
Ultrafast	81.8000	−2.5960	0.0107

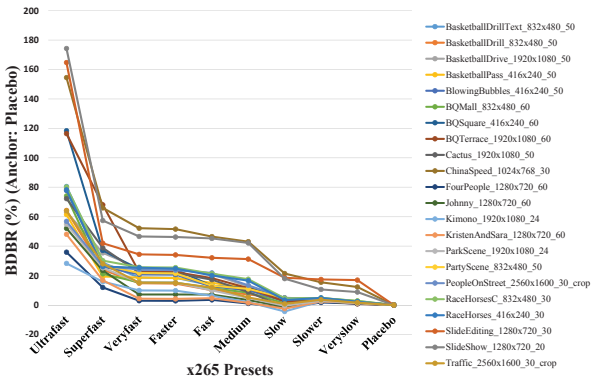

Fig. 1. Performance Comparison of x265 Default Presets

A. x265 Default Presets

x265 provides empirically 10 default presets, which are analyzed in terms of Bjøntegaard Delta Bit Rate (BDBR), Bjøntegaard Delta PSNR (BDPSNR) [12] and normalized time in this paper. Normalized time can be calculated according to equation (2). The encoding time and RD performance of the default presets are shown in TABLE I and Fig. 1, where two unexpected phenomena are observed. Firstly, the RD performance of Slow is better than Slower in spite of less encoding time. Secondly, Faster and Veryfast have similar performance. These verify the defects of empirically determined parameters, which have to be removed for efficient application. Besides, from TABLE I, we also find that Slow is an ideal preset for practical use with the following advantages:

- The Slow preset acts as the critical point of RD performance and encoding time.
- Considering the encoding time of Slow is only 10.63% of Placebo, its performance degradation is relatively acceptable with loss of 4.14% BDBR and $-0.28dB$ BDPSNR.

By contrast, presets faster than Slow bring about dramatic performance drop. In fact, Slow and Medium are used in most practical applications. Therefore, optimization is conducted based on the Slow preset in this paper.

B. Preset-Related Parameters

All the parameters that vary from one default preset to another are listed in TABLE II, where those of Slow are shown in bold. As modification of parameter value will lead to different encoding time and RD performance, all parameter values from TABLE II have to be exhausted to obtain the best parameter configuration. But notice as many as 983040 ($2^{12} \times 3 \times 4^2 \times 5$) combinations are provided by TABLE II. Therefore, an analysis model has to be designed for simplifying the parameter choosing process.

TABLE II
RELATED PARAMETERS IN DEFAULT PRESETS (SLOW TO ULTRAFAST)

"Rect" enables analysis of rectangular motion partitions and "SubpelRefine" specifies amount of subpixel refinement to perform.

Parameter	Values	Parameter	Values
Rect	**1**,0	Ref (frame)	**4**,3,2,1
Lookahead Depth	**25**,20,15,10,5	Lookahead Slices	**4**,8
Bframes	**4**,3	FastIntra	**0**,1
RDlevel	**4**,3,2	Earlyskip	**0**,1
RDOQlevel	**2**,0	WeightPred	**1**,0
SubpelRefine	**3**,2,1,0	SAO	**1**,0
MergeCandidates	**3**,2	Max CU	**64**,32
SearchMethod	**STAR**,DIA	Min CU	**8**,16

III. PARAMETER SELECTION MODEL

In this section, an analysis model is designed for choosing parameters and adapting parameter values which perform better than the default presets. Firstly, a criterion is presented for parameter assessment. Then, important parameters are picked out using the criterion. Finally, an MCMC based algorithm is proposed to obtain appropriate configurations.

A. RDT Criterion

Since parameter variation results in different encoding time and RD performance, RDTscore (Rate-Distortion-Time score) is adopted for parameter assessment

$$RDTscore = \frac{TS}{BDBR} = \frac{1 - Time(Normalized)}{BDBR} \quad (1)$$

where TS denotes time saving, and the normalized time is calculated with respect to the anchor preset.

$$Time(Normalized) = \frac{EncodingTime}{EncodingTime(AnchorPreset)} \quad (2)$$

Note that in Section II.A the anchor preset is Placebo while in Section III and IV the anchor preset is Slow.

Higher RDTscore represents higher speed-up ratio at the same RD performance loss, or less RD performance loss at the same speed up ratio.

B. Coarse-grained Parameter Screening

To evaluate the influence of parameters on time and RD performance, we tune only one parameter value in TABLE II per time based on the Slow preset. The resultant encoding Time, BDBR, BDPSNR and RDTscore of each parameter value are listed in TABLE III.

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

TABLE III
RDT PERFORMANCE OF ALL PARAMETERS

ParamSet	BDBR	BDPSNR	Time (Normalized)	TS	RDTscore	Class
RDlevel = 3	-0.2603	0.0084	0.9889	0.0111	-0.0425	**1**
SubpelRefine = 2	-0.2248	0.0061	0.9070	0.0930	-0.4136	**1**
Ref (frame) = 3	-0.0262	-0.0012	0.9872	0.0128	-0.4909	**1**
SubpelRefine = 1	-0.0716	-0.0001	0.9014	0.0986	-1.3785	**1**
RDOQlevel = 0	1.3694	-0.0518	0.7871	0.2129	0.1555	**2**
CUsize = 32~8	2.6636	-0.0990	0.6102	0.3898	0.1463	**2**
EarlySkip = 1	1.4206	-0.0550	0.8859	0.1141	0.0803	**2**
Rect = 0	3.1891	-0.1523	0.7881	0.2119	0.0664	**2**
Ref (frame) = 1	2.4342	-0.0902	0.8412	0.1588	0.0653	**2**
SearchMethod = DIA	1.2330	-0.0832	0.9220	0.0780	0.0632	**2**
Ref (frame) = 2	0.8687	-0.0412	0.9534	0.0466	0.0537	**2**
SubpelRefine = 0	3.3198	-0.1273	0.8534	0.1466	0.0442	**2**
RDlevel = 2	4.7943	-0.1801	0.7967	0.2033	0.0424	**2**
CUsize = 32~16	20.5701	-0.9829	0.4011	0.5989	0.0291	**2**
Lookahead Depth = 20	-0.2864	0.0096	1.0121	-0.0121	0.0422	**3**
Lookahead Depth = 15	-0.2864	0.0096	1.0092	-0.0092	0.0322	**3**
Lookahead Depth = 10	-0.2864	0.0096	1.0062	-0.0062	0.0217	**3**
CUsize = 64~16	16.7265	-0.8474	0.6705	0.3295	0.0197	**3**
Lookahead Slices = 8	-0.2907	0.0098	1.0007	-0.0007	0.0024	**3**
SAO = 0	2.8925	-0.1260	1.0022	-0.0022	-0.0008	**3**
WeightPred = 0	-0.2834	0.0090	0.9996	0.0004	-0.0013	**3**
MergeCandidates = 2	0.1919	-0.0057	1.0003	-0.0003	-0.0016	**3**
FastIntra = 1	-0.2080	0.0062	0.9979	0.0021	-0.0103	**3**
Lookahead Depth = 5	-0.2864	0.0096	0.9960	0.0040	-0.0141	**3**
Bframes = 3	1.0614	-0.0412	1.2367	-0.2367	-0.2230	**3**

TABLE IV
SELECTED PARAMETERS

Parameter	Values	Parameter	Values
Rect	**1**,0	Ref (frame)	**3**,2,1
SubpelRefine	**2**,0	Max CU	**64**,32
SearchMethod	**STAR**,DIA	Min CU	**8**,16
RDlevel	**3**,2	Earlyskip	**0**,1
RDOQlevel	**2**,0		

According to the impact on time and RD performance, parameter values are classified into three classes, as shown in TABLE III. The parameter values which are beneficial for both reducing encoding time and improving RD performance are classified into Class 1. The parameter values which have obvious influence on RDT performance are classified into Class 2. The parameter values which either have negligible effects on encoding time and RD performance or have too small RDTscore are classified into Class 3. The parameter values in Class 1 have excellent properties, so they are applied to replace the parameter values in the Slow preset, which lays a good foundation for the following optimization. The parameter values in Class 2 enable complexity scalability for encoders while those in Class 3 are filtered out from parameter options.

According to the results of parameter screening, TABLE II is updated to TABLE IV, where only important parameters or parameter values are shown and initial setups are modified with parameter values in Class 1.

C. MCMC Based Parameter Adaptation

The parameters in TABLE IV correlate closely with encoding time and RD performance and their RDTscore can be found in TABLE III. In some cases, several parameters have to be combined for a given encoding time. Considering the interaction between parameters, all combinations of parameters have to be exhausted to acquire the optimal one. But due to the large number of parameter combinations, in this paper an MCMC based algorithm as shown in Algorithm 1 is proposed for efficient parameter adaptation in a practical way.

Algorithm 1: MCMC based Parameter Adaptation

Input: Target Encoding Time: T_t,
Real Encoding Time using Param: $T_r(Param)$,
Initial Parameter Configuration: $Param_{init}$,
Time Error Tolerance: TE,
Loop Times: N_{loop1},N_{loop2}
Output: Selected Parameter Configuration: $Param_{best}$

$clock = 0$;
$Param_{best}, Param_{this} = Param_{init}$;
while $(||T_t - T_r(Param_{best})|| > TE) or (clock \leq N_{loop1})$ **do**
  $Param_{this} = Param_{best}$;
  $i = random(1, Num(Param_{init}))$
  $Param_{this}(i).value$ = randomly new one;
  if $(||T_t - T_r(Param_{this})|| \leq ||T_t - T_r(Param_{best})||)$ **then**
    $Param_{best} = Param_{this}$;
  else if $(RDTscore(Param_{this}) > RDTscore(Param_{best})) \&\& (||T_t - T_r(Param_{this})|| \leq TE)$ **then**
    $Param_{best} = Param_{this}$;
  $clock = clock + 1$;
  Encode with $Param_{best}$, update T_r;
$clock = 0$;
while $clock \leq N_{loop2}$ **do**
  $Num_{ParamChange} = random(1, Num(Param_{init}))$
  for i *in* $Num_{ParamChange}$ **do**
    $Param_{this}(i).value$ = randomly new one;
  if $(RDTscore(Param_{this}) > RDTscore(Param_{best})) \&\& (||T_t - T_r(Param_{this})|| \leq TE)$ **then**
    $Param_{best} = Param_{this}$
    Encode with $Param_{best}$, update T_r and $RDTscore$;
  $clock = clock + 1$;
return $Param_{best}$;

The basic idea of the algorithm is to iteratively generate a new parameter configuration from the current configuration and determine whether the new one is better. The algorithm is composed of two loops, which place emphasis on encoding time restriction and RDT performance optimization respectively. In both loops, best parameter configuration will be updated to a generated one under given conditions as shown in Algorithm 1. With this parameter adaptation algorithm, a parameter configuration close to the best performance at a target encoding time can be acquired without exhaustive parameter combinations.

IV. EXPERIMENTS AND ANALYSIS

The experiments are based on x265 (2.3) with GOP 30 and carried out in a workstation with Intel Xeon E5-4627 v3, 2.6 GHz, 20 cores and 128 GB RAM Memory. All sequences from Class A to F are tested for each configuration and each QP (22, 27, 32 and 37).

To verify the effectiveness of the proposed parameter adaptation algorithm, all 768 ($2^8 \times 3$) combinations of parameter

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

TABLE V
PARAMETER CONFIGURATIONS GENERATED BY PROPOSED ALGORITHM

ParamSet	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Rect	1	1	1	1	1	1	1	0	1	0
RDlevel	3	3	3	3	3	3	3	2	2	2
RDOQ	2	2	0	0	2	0	0	0	0	0
SubpelRefine	2	2	2	2	2	2	2	2	0	2
EarlySkip	0	0	0	0	0	0	0	0	1	0
max CU	64	64	64	64	32	32	32	32	32	32
min CU	8	8	8	8	8	8	8	8	16	16
SearchMethod	STAR	STAR	STAR	STAR	STAR	STAR	DIA	STAR	DIA	DIA
Ref (frame)	3	1	3	2	3	3	3	2	3	3
Target Encoding Time	0.9000	0.8000	0.7000	0.6000	0.5000	0.4000	0.3000	0.2000	0.1500	0.1000
Real Encoding Time	0.8902	0.7651	0.6574	0.6206	0.5053	0.3903	0.3037	0.2391	0.1813	0.1281
BDBR (%)	0.0249	2.5955	1.6692	2.5915	1.9301	3.5821	8.3380	18.4732	38.9091	67.1615
BDPSNR (dB)	-0.0008	-0.1334	-0.0857	-0.1321	-0.0996	-0.1827	-0.4158	-0.8536	-1.5214	-2.4831

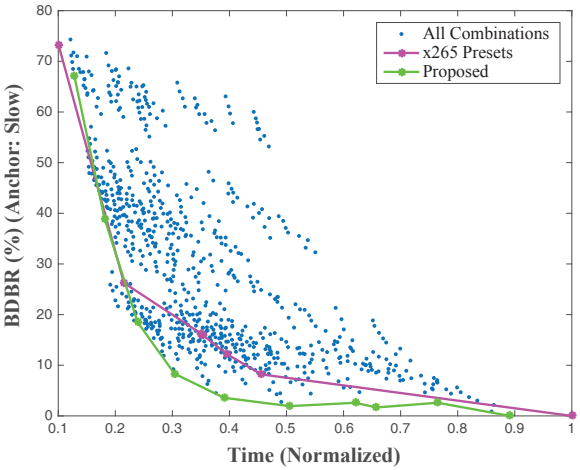

Fig. 2. BDBR-Time of Parameters with respect to the Slow Preset

values in TABLE IV are applied to x265 at first. The resultant encoding time and BDBR are shown as splashes in Fig. 2, where each point in this figure corresponds the average value of the 10 runs for each test sequence and the lower envelope of splashes gives the optimal results of parameter adaptation. Then, by Algorithm 1, parameter configurations are generated for 10 representative normalized target time as shown in TABLE V. In the Algorithm 1, TE is set as 0.05. $Param_{init}$ represents parameters shown in bold in TABLE IV. N_{loop1} and N_{loop2} are 50 and 200 respectively. The corresponding performance of each configuration is shown both in TABLE V and Fig. 2. Furthermore, the RDT performance of x265 default presets is also illustrated in Fig. 2 for comparison.

As shown in Fig. 2, configurations obtained by the proposed model can well approximate the optimal envelope in spite of relative low complexity of parameter selection. What's more, they also perform much better than x265 default presets at high target time. To evaluate the performance gain of the proposed model at high target time, we calculate the difference in a curve fitting manner [12] by selecting 4 pairs of configurations with close BDBR as shown in TABLE VI.

TABLE VI
CONFIGURATION PAIRS BASED PERFORMANCE EVALUATION

x265 Default Presets	BDBR (%)	Time (Normalized)	Generated Configurations
Slow	0.0000	1.0000	P1
Medium	8.1922	0.4562	P5
Fast	12.2018	0.3954	P7
Faster	15.9919	0.3530	P8
Average Time Saving			-26.72%

Average 26.72% encoding time is saved while maintaining similar RD performance to x265 default presets.

The reason why only minor gain than the default presets is obtained at low target time is that parameters only have a limited number of valid values at this time, which leaves little space of optimization for the proposed model. Nevertheless, considering such short time is not expected by many applications, the proposed model can still obtain obvious gains in most situations.

V. CONCLUSION

This paper presents an analysis on the performance of x265 encoder, followed by a comprehensive analysis on the effects of x265 parameters. Important parameters are picked out based on the analysis. An MCMC based algorithm is then proposed for efficient parameter adaptation at a target encoding time. The model is proved to be able to approximate optimal parameter values without exhausting the parameter combinations. Experimental results show that average 26.72% time reduction can be achieved while maintaining similar RD performance to x265 default presets using the proposed algorithm. Due to its effectiveness and universality, the proposed parameter selection model can be also applied to other encoders.

ACKNOWLEDGMENT

This work was supported by NSFC (61671296, 61521062, U1611461 and 61601282), the 111 Project (B07022 and Sheitc No.150633) and the Shanghai Key Laboratory of Digital Media Processing and Transmissions.

REFERENCES

[1] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, Dec 2012.

[2] S. W. Falei Luo, Siwei Ma, "Overview of the second generation avs video coding standard (avs2)," http://wwwen.zte.com.cn/endata/magazine/ztecommunications/2016/1/articles/201603/t20160311_448968.html, accessed March 11, 2016.

[3] HM.HEVC Test Model [Online].Available: http://hevc.hhi.fraunhofer.de/svn/svnHEVCSoftware/.

[4] G. Clare, F. Henry, and S. Pateux, "Wavefront parallel processing for hevc encoding and decoding," *document JCTVC-F274. Torino, Italy, Jully*, 2011.

[5] M. B. Cassa, M. Naccari, and F. Pereira, "Fast rate distortion optimization for the emerging hevc standard," in *Picture Coding Symposium (PCS), 2012*. IEEE, 2012, pp. 493–496.

[6] L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An effective cu size decision method for hevc encoders," *IEEE Transactions on Multimedia*, vol. 15, no. 2, pp. 465–470, 2013.

[7] G. Corrêa, P. A. Assunção, L. V. Agostini, and L. A. da Silva Cruz, "Pareto-based method for high efficiency video coding with limited encoding time," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1734–1745, 2016.

[8] G. Correa, P. Assuncao, L. Agostini, and L. A. da Silva Cruz, "Performance and computational complexity assessment of high-efficiency video encoders," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1899–1909, 2012.

[9] M. Grellert, S. Bampi, and B. Zatt, "Complexity-scalable hevc encoding," in *2016 Picture Coding Symposium (PCS)*, Dec 2016, pp. 1–5.

[10] x265.x265 Software [Online].Available: https://bitbucket.org/multicoreware/x265/downloads/.

[11] C. M. Carlo, "Markov chain monte carlo and gibbs sampling," *Lecture notes for EEB*, vol. 581, 2004.

[12] G. Bjontegaard, "Calcuation of average psnr differences between rd-curves," *Doc. VCEG-M33 ITU-T Q6/16, Austin, TX, USA, 2-4 April 2001*, 2001.

978-1-5386-4881-0/18/$31.00 ©2018 IEEE