TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

MARKO VIITANEN

HEVC PARAMETER EXPLORATION FOR EFFICIENT MODE DECISION

Master of Science thesis

Examiner: Ass. Prof. Jarno Vanne
Examiner and topic approved by the
Faculty Council of the Faculty of
Computing and Electrical Engineering
on 27th September 2017

ABSTRACT

MARKO VIITANEN: HEVC Parameter Exploration for Efficient Mode Decision
Tampere University of Technology
Master of Science thesis, 49 pages, 3 Appendix pages
October 2017
Master's Degree Programme in Information Technology
Major: Embedded Systems
Examiner: Ass. Prof. Jarno Vanne
Keywords: HEVC, Video Compression, Analysis, Mode Decision, Software Development

High Efficiency Video Coding (HEVC) is the latest video coding standard. It halves the achieved bit rate compared with the previous standard, Advanced Video Coding (AVC). However, the bit rate decrease comes with 40% increase in encoding complexity. This is mainly due to larger number of block coding modes, including Symmetric motion partitions (SMPs), Asymmetric motion partitions (AMPs), and larger coding units of up to 64×64 pixels. These new features are mainly used for Inter prediction that accounts for 60-70% of the whole encoding time. For this reason, optimization of Inter prediction is the main topic in this Thesis.

To tackle the Inter prediction complexity, a parametric exploration was chosen as the approach. The exploration was done by gradually shifting the focus from the most coarse optimization to the parameter fine tuning. The selected approach in this study required thousands of individual tests so an automated solution was needed. This led to the creation of a new software solution, TUT Task Manager. It is capable of automatically distributing the tasks of parametric exploration to any number of nodes available in the local network. In total, TUT Task Manager was used to run 4000 tests with a combined CPU time of 14 months.

The results were used to create a set of recommended schemes for Inter mode selection. Overall, these new schemes are shown to provide 31-50% complexity saving against the default configuration of HM 11.0, with a minor bit rate increase of 0.2-1.3%. They also provide better RDC performance than the existing solutions. The tools and methods used in this work are so generic that they can be used to further optimize other parts of the video codec.

TIIVISTELMÄ

MARKO VIITANEN: Parametrien etsintä HEVC:n tehokkaalle moodivalinnalle
Tampereen teknillinen yliopisto
Diplomityö, 49 sivua, 3 liitesivua
Lokakuu 2017
Tietotekniikan koulutusohjelma
Pääaine: Sulautetut Järjestelmät
Tarkastajat: Ass. Prof. Jarno Vanne
Avainsanat: HEVC, Videonpakkaus, Moodinvalinta, Ohjelmistokehitys

High Efficiency Video Coding (HEVC) on tuorein videonpakkausstandardi. Verrattuna aiempaan Advanced Video Coding (AVC) standardiin, HEVC kykenee puolittamaan tarvittavan bittinopeuden samalla laadulla. HEVC vaatii kuitenkin huomattavasti enemmän laskentatehoa, mikä johtuu suuresta määrästä uusia lohkonkoodausmoodeja, kuten symmeriset/asymmetriset lohkot ja 64×64 pikselin kokoiset koodauslohkot. Näitä uusia ominaisuuksia hyödynnetään pääasiassa Inter-koodauksessa, joka kuluttaa 50-70% koko laskenta-ajasta. Siksi Inter-koodaus on valittu tämän työn pääaiheeksi.

Tässä työssä Inter-koodauksen analysointi ja optimointi perustuvat koodausparametrien säätämiseen. Käytännössä parametrien tutkiminen aloitettiin korkean tason parametrimuutoksilla ja tuloksien perusteella siirryttiin kohti parametrien hienosäätöä. Tässä lähestymistavassa tarvittiin tuhansia testiajoja, joiden suorittamiseksi oli kehitettävä uusi ohjelmistoratkaisu nimeltään TUT Task Manager. TUT Task Manager pystyy ajamaan automaattisesti testejä jakaen niitä verkon yli tietokoneille. Yhteensä testejä ajettiin yli 4000 ja testaukseen kului prosessoriaikaa noin 14 kuukautta.

Testituloksien perusteella koostettiin lista suositelluista parametreista Inter-moodivalintaan. Verrattuna HM 11.0 referenssitoteutuksen oletusparametreihin nämä uudet parametrisoinnit vähentävät laskentatehovaatimuksia 31-50% ja nostavat bittinopeutta 0.2-1.3%. Esitetyt tulokset ovat parempia kuin aiemmilla optimointitoteutuksilla. Työssä kehitetyt työkalut ja metodit ovat yleiskäyttöisiä ja niitä voidaan jatkossa käyttää myös muiden videokoodekin osien optimointiin.

PREFACE

This Master of Science Thesis was written in the Laboratory of Pervasive Computing at Tampere University of Technology as a part of research.

Biggest thanks go to my examiner Jarno Vanne, who also acts as my supervisor, for the guidance and encouraging words provided during the process of this Thesis. I would also like to thank my co-workers Joni Räsänen and Ari Lemmetti for the fruitful discussions about my Thesis. I am also grateful for the other Ultra Video Group members for supporting me with the process.

Special thank goes to my girlfriend Thuy for providing the needed support during the final phase of writing.

Tampere, 25.10.2017

Marko Viitanen

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND SYMBOLS

AMP	Asymmetric motion partition
AMVP	Advanced Motion Vector Prediction
ASR	Adaptive Search Range
AVC	Advanced Video Coding
BD-rate	Bjøntegaard Delta bit rate
BSD	Berkeley Software Distribution
CB	Coding Block
CBF	Coded Block Flag
CFM	CBF Fast Mode
CTB	Coding Tree Block
CTU	Coding Tree Unit
CU	Coding Unit
ECU	Early CU
EPD	Early Partition Decision
ESD	Early Skip Detection
HEVC	High Efficiency Video Coding
HiP	High Profile
HM	HEVC Test Model
JCT-VC	Joint Collaborative Team on Video Coding
LB	Low-delay B
MCpF	Million Cycles per Frame
MP	Main Profile
MPEG	Moving Picture Experts Group
PSNR	Peak Signal-to-Noise Ratio
QP	Quantization Parameter
RA	Random Access
RD	Rate-Distortion
RDC	Rate-Distortion-Complexity
SAO	Sample Adaptive Offset
SMP	Symmetric motion partition
TB	Transform Block
TU	Transform Unit
TUT	Tampere University of Technology
VCEG	Video Coding Experts Group

1. INTRODUCTION

Digital video compression became relevant in the 1980s with an introduction of the first video coding standard, H.120 in 1984 [1] . Since then, the compression efficiency has continuously improved by the introduction of a series of standards such as H.261 (1990), H.262 / MPEG-2 (1994), H.263 (1995), H.264 / MPEG-4 Part 10 / *Advanced Video Coding* (AVC) [2] in 2003 and finally the latest H.265 / MPEG-H part 2 / *High Efficiency Video Coding* (HEVC) [3] in 2013.

The research done for the Thesis dates back to 2012-2014 when HEVC standard was being finalized and practical implementations of the standard were lacking. New features made HEVC reference encoder up to 3.2× as complex as AVC reference implementation [4]. The increased complexity of HEVC raises questions on the possibility for optimized solution that still hold most of the benefits of the new tools but with significantly reduced complexity.

Other optimization techniques are available, but they provide no overall solution for *quantization parameter* (QP)-independent and QP-specific cases. This work presents a solution for those cases. The scope of this Thesis is limited to the selection of efficient HEVC coding modes, including the software tool and the exploration environment used for completing this task. The main findings of this work have been published in the IEEE Transactions on Circuits and Systems for Video Technology in 2014 [5].

This Thesis is arranged as follows: Chapter 2 gives an introduction to the HEVC video coding standard. It also introduces available software solutions for HEVC encoding. Chapter 3 presents TUT Task Manager that is a tool for running and distributing the test results to a network of computers. Chapter 4 presents the main contribution of this work. It includes the HEVC parametric exploration, which is used for defining the new Inter mode decision schemes. Comprehensive comparison with the previous art is also presented in this chapter. Chapter 5 gives general guidelines for using the proposed schemes. Finally, Chapter 6 concludes the Thesis.

2. HEVC VIDEO STANDARD

HEVC is the latest of international video coding standards and it is designed for next-generation video contents with transmission and storage efficiency in mind. HEVC is a result of joint collaboration between ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG) as the Joint Collaborative Team on Video Coding (JCT-VC). In January 2013, the final draft of HEVC standard was approved with three profiles [3]: Main Profile (MP), Main 10 Profile (10 bit coding), and Main Still Picture Profile. In this Thesis, the analysis of HEVC encoder focuses on Main Profile.

As a next-generation standard, HEVC provides a significant boost in coding performance over that of the AVC standard [6]. Compared with AVC High Profile (HiP), HEVC reference software can achieve nearly 40% bit rate reduction [4] with similar picture quality when all essential coding tools are enabled. Subjective visual quality metrics have shown even higher bit rate savings of up to 50% [7]. Low-delay, low bit rate, and high resolution applications can get the best of HEVC efficiency. However, the respective complexity is increased by around 1.4× [4].

Prior video coding standards since H.261 use the well-known hybrid video coding scheme that is composed of Inter/Intra prediction, transform coding, and entropy coding. HEVC adopts the same scheme, but extends [8] the traditional 16×16 pixel *macroblocks* (MBs) to a scheme supporting block sizes from 64×64 pixels down to 4×4 pixels in a block tree arrangement [3] [9]. This new scheme is used to adapt from highly textured to large homogeneous regions of the picture. It makes use of more content-adaptiveness in the compression domain.

Majority of the coding gain of HEVC comes from the wider range of possible block structures, which also introduces the majority of the computational overhead. Rate-distortion (RD) in Inter coding is particularly complex due to the numerous prediction modes and block partitions to evaluate [8].

2.1 Block structure

In HEVC, pictures are partitioned into basic processing units called *coding tree units* (CTUs) [3]. Each CTU is made of three *coding tree blocks* (CTBs): one square CTB of luma samples and two corresponding CTBs of chroma samples. CTBs also contain the associated syntax elements. The size of the luma CTB is defined as $2N_{MAX}\times 2N_{MAX}$ [8], where $N_{MAX} \in \{8, 16, 32\}$. Selection of N_{MAX} is made by the encoder depending on the different computational and memory demands.

2.1.1 Coding units

The coding units in HEVC are arranged in a quadtree structure where CTU is used as a root node (at depth $h = 0$) and CTBs can be partitioned into smaller *coding blocks* (CBs). CTBs (of a CTU) can be optionally divided into four equal-size square CBs at depth $h + 1$. The quadtree can be futher divided recursively, until the maximum depth h_{MAX} in the hierarchy is reached. Adjustable quadtree depth is one of the tools to handle content-adaptive block granularity in HEVC.

The quadtree leaf nodes are called *coding units* (CUs) [8]. For each CU, the size of its luma CB can be defined as $2N\times 2N$, where $N \leqslant N_{MAX}$. In HEVC, $N \in \{4, 8, 16, 32\}$, so $N_{MIN} = 4$ and $h_{MAX} = 4$. The CUs of the coding tree are processed in a depth-first manner. A CU defines a picture region that shares the same prediction mode (Intra, Inter, Skip or Merge), thus acting as a root for a prediction tree as well as a root for a transform tree [3].

2.1.2 Prediction units

A *prediction unit* (PU) is composed of adjacent luma (and chroma) *prediction blocks* (PBs) that share the same prediction process and an identical prediction information signaled in the bit stream. HEVC prediction tree allows luma/chroma CBs to be divided into one, two, or four rectangular-shape PBs. *Symmetric motion partition* (SMP) and *Square motion partition* (Square) are adopted from AVC and extended to support larger CB sizes. *Asymmetric motion partition* (AMP) modes are introduced in HEVC for irregular shapes that would take more bits to code with Square and SMP modes.

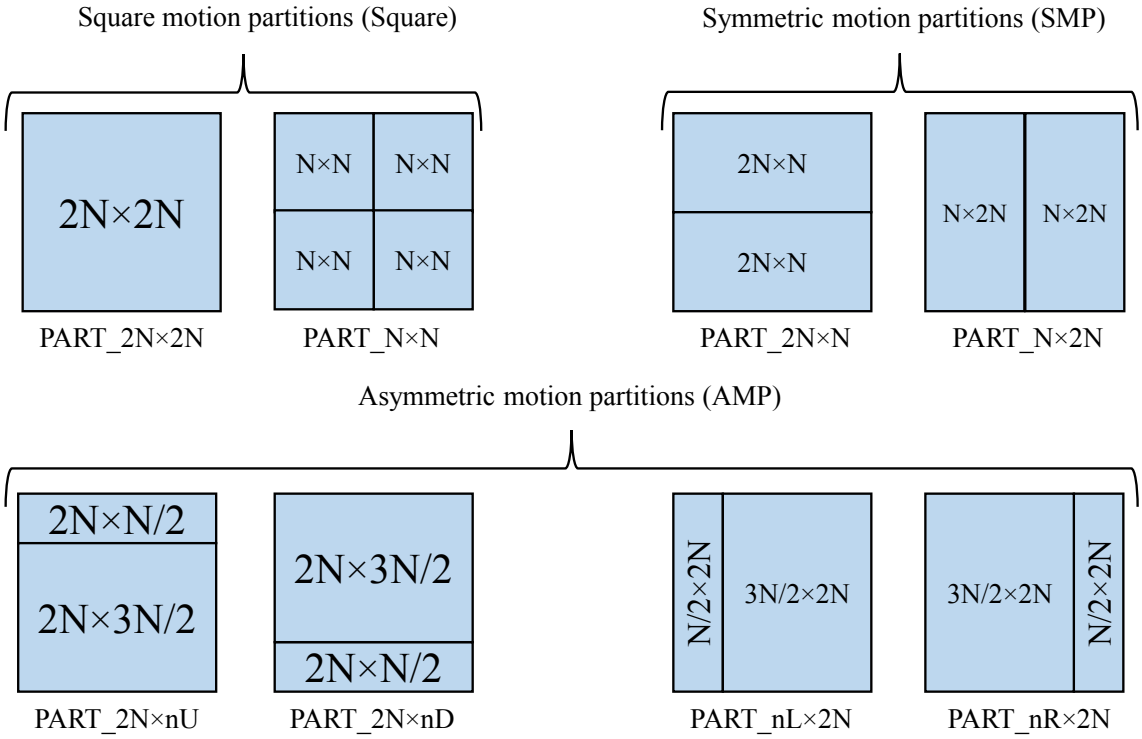


Figure 2.1 *HEVC inter prediction partition modes*

Figure 2.1 illustrates the partition modes and associated luma PB shapes specified for the Inter-coded CBs of size $2N \times 2N$. The supported mode set includes two Square modes (PART_2N×2N and PART_N×N), two SMP modes (PART_2N×N and PART_N×2N), and four AMP modes (PART_2N×nU, PART_2N×nD, PART_nL×2N, and PART_nR×2N) [8]. With the AMP modes, n represents smaller one of the partitions, whereas U, D, L, and R denote up, down, left, and right, respectively. For example nU denotes that the smaller partition is in the upper part of the block. The use of CU as the root for prediction guarantees that the size of a PB is always the same or smaller than that of CB.

The following limits are set for PBs in HEVC standard:

1. $PART\_N\times N$ mode can only be used at depth h_{MAX} for Inter-coded CBs, since the recursive quadtree splitting (division to four quadrants) gives an equal result with $PART\_2N\times 2N$ mode when $h < h_{MAX}$.

2. The smallest PBs of 4×4 pixels are disabled in Inter coding to minimize the worst-case memory bandwidth in HEVC. That is, $PART\_N\times N$ can be entirely omitted in the HEVC Inter prediction when $N_{MIN} = 4$.

3. Block sizes smaller than 4×4 are not allowed, thus disabling the use of AMP modes when $N = 4$.

4. Skip mode is always $PART\_2N{\times}2N$ mode and Intra predictions can only be the Square modes.

HEVC integrates a a leaf merging concept [10] to improve Inter prediction further. Enabling the use of Merge mode, PBs of neighboring PUs, in pre-defined locations, sharing the identical motion information can be conceptually merged into larger region. In this case, each PB only signals the identification number (from five candidates) to copy the motion info from. Since only the prediction error needs to be coded individually for each PB, the coding cost is reduced. Special case of the Merge mode, the Skip mode, signals that no prediction error is coded for it, thus saving even more on coding cost.

2.1.3 Transform units

A *transform unit* (TU) specifies a picture region (residual block) sharing the same transformation. Starting from a CU (leaf nodes) of a coding tree, transform tree with *transform blocks* (TBs) for luma and chroma components forms a nested quadtree. TB is limited to square sizes of 4×4, 8×8, 16×16, or 32×32 so that size is at maximum that of CU. Further TU analysis is out of the scope of this Thesis.

2.2 Coding tools

Compression efficiency of the standard is dependent on the available tools to use for predictions, filtering, and parallelization. HEVC standard defines only the methods to decode such info. Hence, it is up to the encoder to decide how to use the available tools.

2.2.1 Intra prediction

Intra prediction only uses data from the current frame being encoded. Because the coding order is from top to bottom and left to right, only the top and left blocks are available for prediction. HEVC uses 35 different Intra prediction modes for luma

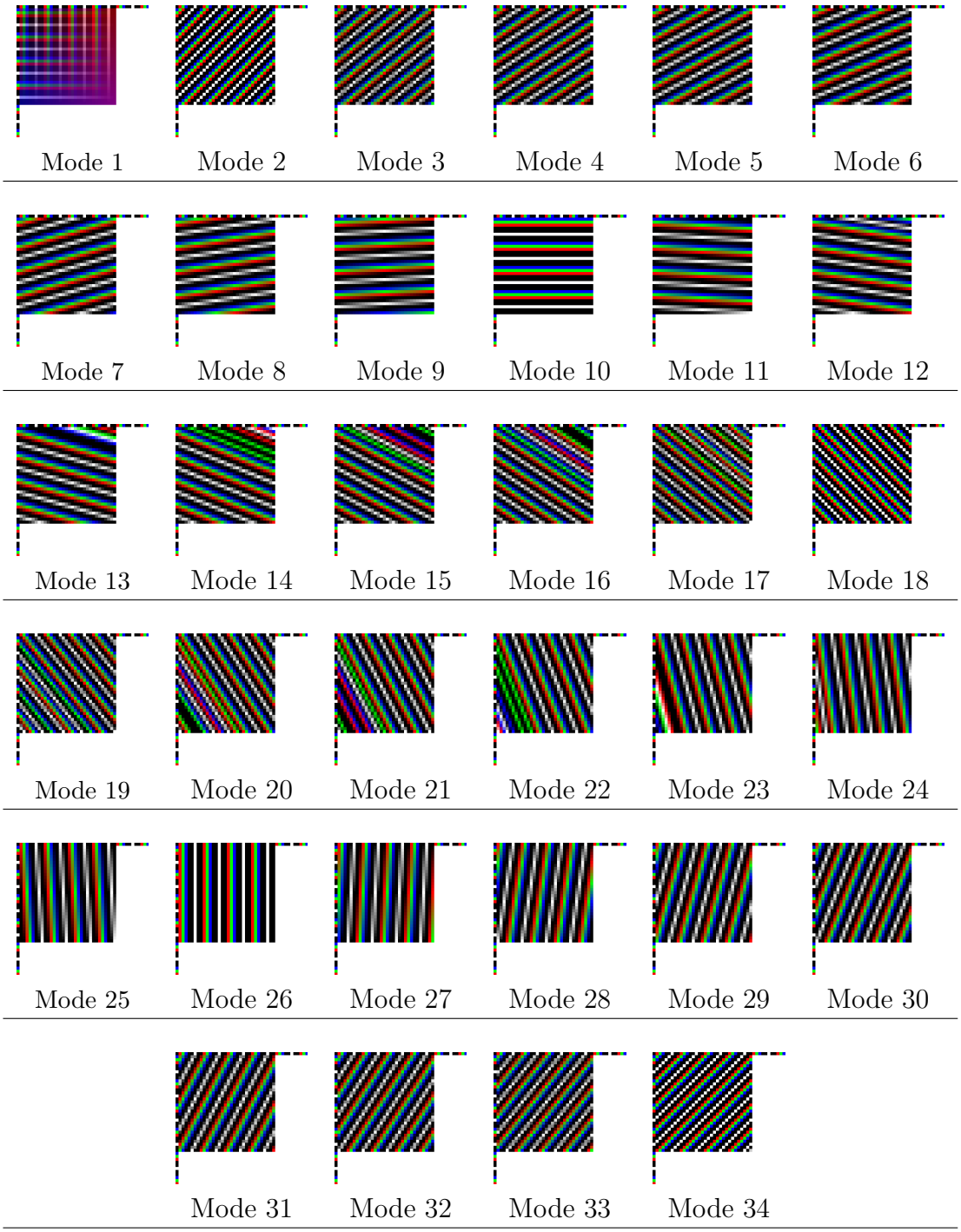


Figure 2.2 *Intra prediction modes used in HEVC standard.*

prediction, which enhances the available predictions from 9 modes in AVC standard by including more intermediate angles [3].

Intra prediction modes (excluding DC mode) are presented in figure 2.2. They are generated using HEVC Intra prediction functions with repeating pattern $color \in \{Black, Red, Green, Blue, White\}$ as the border reference pixels. This visualizes the blending of neighboring pixel values with different prediction modes.

2.2.2 Inter prediction

Inter coding uses other frames as references. A CB can include signaling of motion info in order to use the block of equal size from one of the references [3]. [1]

Motion Vectors (MVs) signal the position offset to extract the reconstructed pixels from a reference frame. *Advanced Motion Vector Prediction* (AMVP) is one of the specified methods for decreasing bit coding cost in HEVC. AMVP selects two spatial candidates from left and top as well as one temporal candidate. The selected candidate acts as the starting point for the MV, and only the Δ offset is coded to the bit stream. Candidate selection is a complex process and includes MV scaling depending on the reference frame index and other details.

MVs are coded in $\frac{1}{4}$ pixel accuracy by using 8-tap filter for the fractional pixel calculations which is quite computationally intensive operation.

2.2.3 Filtering

HEVC standard specifies two separate filtering algorithms. The deblocking filter is from the previous standards. It filters the border pixels of the CB by simple rules based on the prediction types on border blocks.

A new *Sample Adaptive Offset* (SAO) filter improves coding efficiency by providing a better tool for removing distortions caused by the lossy encoding process. This is done by applying an offset value for each sample in a CTU according to encoder decisions. SAO allows selection between two modes, edge offset and band offset. [3] [11]

[1] Using two reference blocks for one CB is also possible with Bi-prediction mode

2.2.4 Parallelization

Given that HEVC is designed to work with 4K video, two data-level parallelization tools have been included in the standard:

- *Tiles*

  Tiles allow splitting the frame in smaller, rectangular chunks which can be processed in parallel. Prediction across different tiles can be handled by limiting predictions to point to the current tile, but this approach reduces coding efficiency. Another approach is to add logic to force all the neighboring tiles in the reference frames be completed before processing the tile, but this adds dependencies which limits the parallelization.

- *Wavefront Parallel Processing* (WPP)

  WPP uses separate rows of CTUs in encoding, with context sync to next row after two CTUs [3]. This means that the next row can be started when two CTUs of the previous row have been coded. The compression efficiency is only minimally affected and the encoding and decoding can be parallelized fairly effectively. The required context sync causes some limitations to the parallelizations. It requires two CTUs to be coded before next CTU row can be started, so processing can be started with only one thread, increasing as more CTU rows are coded. Additionally, parallelization can be increased further by encoding multiple frames in parallel.

2.3 Software

HEVC reference software is available as open-source and more selection coming available as companies develop their own implementations of HEVC. Most popular open-source encoders are listed in Table 2.1. Since the focus in this Thesis is in encoding, the list is missing available open-source decoders. Multiple proprietary implementations are also available but they are not discussed here.

2.3.1 HM

HEVC Test Model (HM) [12] is a software model used in the development of the HEVC standard. HM has been developed mostly by the companies adding new

Table 2.1 *Overview of the HEVC encoding software available.*

Encoder	License	Developer	Year	Lang.	Encoding uses
HM	BSD	JTC-VC	2010	C++	Research, very high-quality
Homer	LGPLv2.1	Juan Casal	2012	C99	Unknown
Kvazaar	LGPL 2.1	TUT	2012	C99	Live, fast, and high-quality
Turing	GPL 2[1]	BBC	2012	C++11	Memory efficient encoding
x265	GPL 2[1]	MulticoreWare	2013	C++	Live, fast, and high-quality

[1] Commercial license also available

features in the standard, but it is still open-source and distributed with permissive BSD License. The license enables anyone to use the HM code in the other open-source or closed-source software as long HM is mentioned as the source material.

HM contain practically all the HEVC standard features but its performance has not been optimized. Even multi-threading optimizations are lacking and HM can only utilize one CPU core at a time. This means HM cannot be used in the live encoding/decoding cases and is too slow for even normal consumer use.

2.3.2 Homer

One of the newcomers to the HEVC software selection is the HOMER HEVC encoder [13], which is mainly developed by a single person. The encoder includes most of the HEVC features but is still under heavy development.

2.3.3 Kvazaar

Kvazaar HEVC encoder [14] is an open-source software developed by TUT since 2012, meaning the work was started before the HEVC standard was ratified. This fast and portable encoder includes scalable multithreading [15] on multiple platforms [16] [17]. With modern SIMD optimizations [18], it is capable of real-time 4K encoding.

2.3.4 Turing

Turing [19] is actually a complete codec (coder/decoder) unlike the other listed implementations. It is developed by BBC having an ambiguous goal of becoming most

widely used HEVC codec. Development is done using modern C++11 programming language which, however, makes the source code hard to read. Real evidence of the performance is still missing, but this project is an interesting to follow.

2.3.5 x265

x265 [20] is one of the most popular HEVC encoders. It is based on the HM reference software with x264 software encoder tricks included. Altough it is available as an open-source software, GPL license might limit the use in some environments and commercial licenses are available. Because of the dual licensing, all contributors must sign off their rights to the MulticoreWare Inc.

2.3.6 Comparison

Some comparisons between HEVC encoders exist in the literature, but they tend to be limited to Intra-only encoding. In [18], real-time coding capabilities of Kvazaar are compared with x265. All AVX2 and other SIMD optimizations are utilized in both cases. In this evaluation, Kvazaar is found to be 20% faster with 9.1% less bit rate for the same quality.

Since HM does not provide any optimizations, it is not typically compared with other encoders in terms of encoding speed. However, during encoding tool development it is a common practice to implement the changes to HM and compare the modified implementation of HM with the non-modified version. This is also the case in the analysis provided in this Thesis for efficient mode decision.

3. TUT TASK MANAGER

HEVC parameter exploration requires thousands of test runs. Distributing these runs over multiple computers with USB sticks and manually over remote desktop was considered far too time-consuming task. Therefore, an automated solution was needed. Even though closed implementations would have been on the market, it was considered better to create a customized solution for our versatile needs.

Task Manager was designed based on the following requirements:

- Handles data transfer to the clients
- Handles test running automatically
- Fetches and checks the results for completeness
- Keeps communication encrypted if possible
- Keeps only a small taskbar icon visible in clients
- Keeps client quiet and without need for interaction

Qt was chosen as the framework for easier user interface creation and possibility for multi-platform solution.

Figure 3.1 shows the final user interface of the developed TUT Task Manager server. A list of the assigned tasks, status of the tasks, and connected clients are displayed in the same view.

Most of the required test runs were conducted with HM encoder. Although HM is limited to single thread processing, multi-core processors can run multiple instances of the HM at the same time and make use of the whole processing capability.

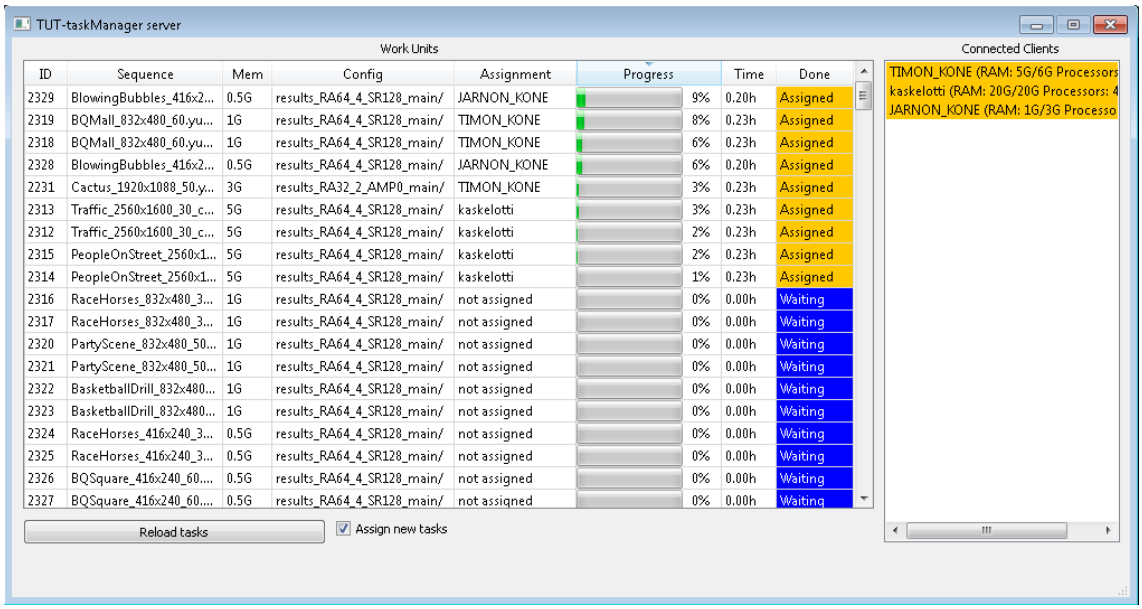


Figure 3.1 *Graphical User Interface of the TUT Task Manager server*

3.1 Features

Although the design of the system is kept simple, some additional features were implemented for better stability. These features are considered next.

3.1.1 Error detection and recovery

Simple error detection scheme was designed to ensure the tasks are completed before submitting the results. A separate script is run in one second intervals to parse the output file of the encoder. The script sends the number of completed frames to the server for progress indication.

When the encoder has completed the task, the same script double-checks that the required number of frames was completed. If this is not the case, client sends the server an error indication.

An error is shown with red task color and manual clicks are required to run it again. This procedure makes sure that a constantly failing task is not consuming the available computation resources.

3.1.2 Input file management

Encoding input is raw YUV 4:2:0 video files with sizes ranging from hundreds of megabytes up to multiple gigabytes. Sending the input files for each task would consume bandwidth whereas keeping the files ready at the client would use a lot of disk space. The input files are therefore kept only if other tasks at the same client need them. Usually tasks needing the same input files are after each other, making the removal of input files at the client side very efficient.

Sending the input files takes some time and connections could be lost before the operation is ready. Therefore, automatic resume of the file transfer is part of the protocol. To make sure the input files have been sent properly, an MD5 hash of the file is verified after receiving it. If the hash does not match the file, the file is sent again until the hash is correct. MD5 hashes of the files are pre-calculated at the server for efficiency.

3.1.3 Automatic updater

Client software was designed to work as a background process, so an automated updating system was implemented. A server checks a version number of the client when the connection is established.

If an old version is detected, an update packet is sent to the client. A packet contains a zip file with a new client program. When the zip file is downloaded to the client, a separate upgrade program is started and the client closed. The upgrade program waits for a couple of seconds, extracts the zip file, and starts the client again.

This simple way of upgrading clients was found to be working in practice without any problems. Only some manual work was required in the server side to prepare for the update, after which the upgrade process is automatic. The needed steps are listed below:

1. A client version number is updated to the server software
2. A new zip packet with the client upgrade is copied to the server folder
3. Server is restarted

4. Any connecting client with an old version is upgraded
5. Tasks are restarted when the client is upgraded (unfinished test runs are lost)

3.2 Protocol

A customized protocol was designed with a simplicity in mind. The protocol is based on packet data including ID followed by the data structure. Appendix A presents a code listing for sending and parsing one packet.

3.3 Future

The TUT Task Manager was designed to handle large batches of test runs and it succeeded in what it was designed for. Adding the following features might make the software usable in other kinds of tasks:

- The designed protocol has flaws with clients parsing packets in serial, since packets do not contain info of the length, which is only found out after parsing the data. The packet could be prepended with the length of the packet.
- Although the used framework is multi-platform, the actual software was built and run only on Windows. Pushing tasks to multiple operating systems would require own encoder and other tool binaries to be built for each operating system.
- Task generation is handled by a different program but could be integrated with TUT Task Manager.
- Current design is based on centralized processing, but it would benefit from the control being distributed.

4. PARAMETER EXPLORATION

To tackle the Inter prediction complexity, a parametric exploration was chosen as the approach. The selected approach in this study required thousands of individual tests so an automated solution was needed. The previously introduced TUT Task Manager is put in use for testing possible modifications for mode decision schemes.

The mode decision process of HM Main Profile (MP) encoder is shown in Fig. 4.1. It is conducted in a recursive fashion for each Inter-coded CU at every quadtree depth, marked as h, after which a final assignment of coding modes is done at CTU level. The presented flowcharts include only the default coding options. A complete flowchart can be found in [21].

In the following, the mode decision process of the HM MP encoder is explained for Inter-coded CU at depth h. First, the HM MP encoder evaluates a CU of size $2N\times2N$ with the Skip and Merge modes. Then, Square and SMP modes are checked for the same CU size and the *best temporary mode* (M'_h) is selected among these five modes. When $N \in \{8, 16\}$, AMP mode decision uses the selected M'_h mode for assigning the next actions (Fig.4.1(a)) as follows:

- if $M'_h \in \{Skip, Merge\}$, no AMP modes are included
- if $M'_h \in \{PART\_2N\times N, PART\_N\times2N\}$, the search goes through only two out of the four AMP modes
- if $M'_h = PART\_2N\times2N$, the search goes through all four AMP modes

After the Intra mode, the *best mode* (M_h) is finally selected among all the evaluated modes.

In HM MP encoder, the AMP mode evaluation is handled with a low-complexity procedure called *Merge mode only testing*, when the conditions of M'_h and the *coding*

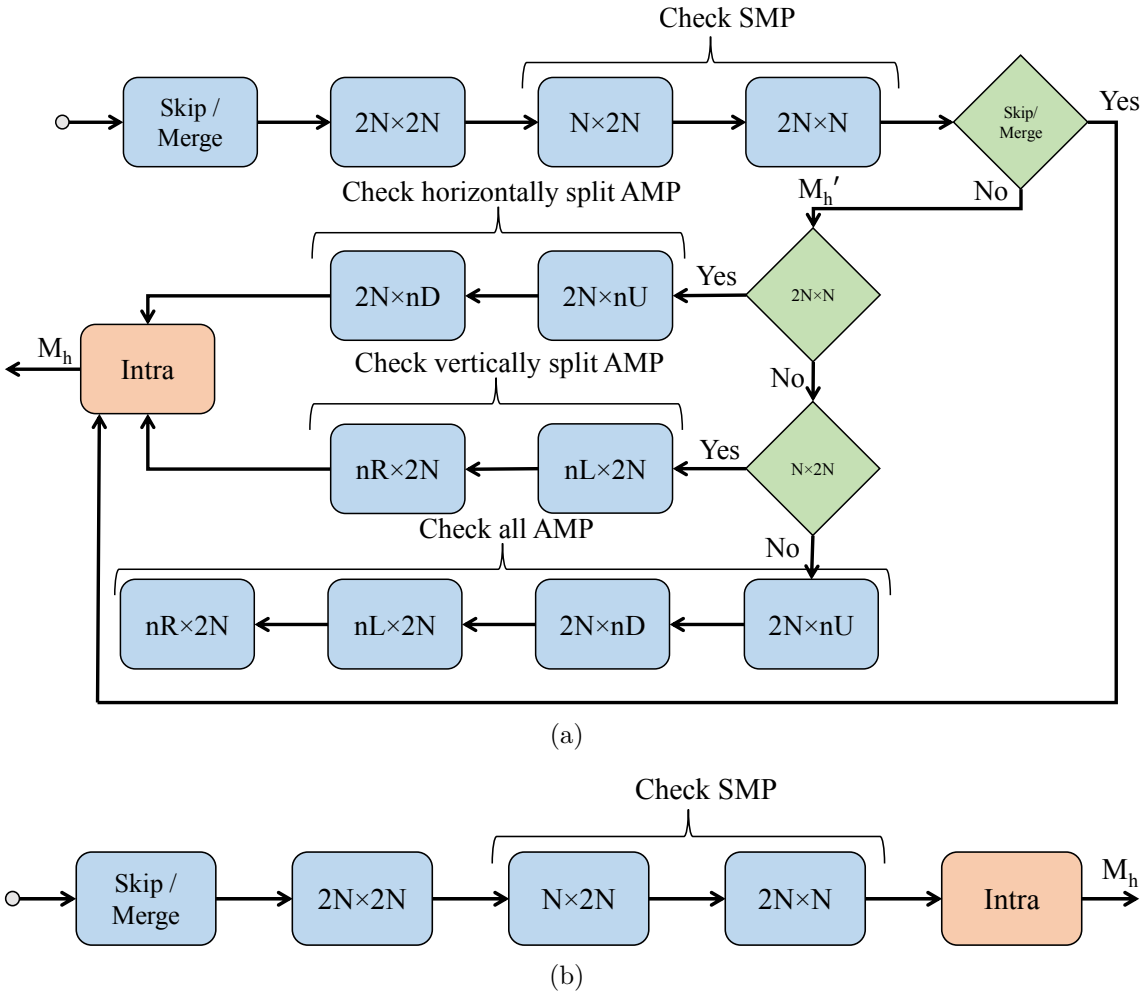


Figure 4.1 *Mode decision in HM encoder. (a) $N \in \{8, 16\}$. (b) $N \in \{4, 32\}$.*

Table 4.1 *Merge-mode only testing conditions in HM encoder.*

Conditions		**AMP Merge modes**			
M_{h-1}	M'_h	$2N{\times}nU$	$2N{\times}nD$	$nL{\times}2N$	$nR{\times}2N$
AMP	-	x	x	x	x
Merge	-	x	x	x	x
Intra	$2N{\times}N$	x	x		
Intra	$N{\times}2N$			x	x

mode of the parent CU (M_{h-1}) match to those in Table 4.1. This Merge mode only testing procedure is omitted in the flowcharts for clarity. When $N = 32$, the AMP mode exploration is limited to the Merge mode only testing, so the flowchart for $N = 32$ is the same as for $N = 4$ (Fig. 4.1(b)).

HM MP also offers three optional early termination mechanisms to speed-up its mode decision:

1. *early CU* (ECU) [22]
   ECU is a simple early termination mechanism that monitors M_h at the end of each search depth h. If $M_h = Skip$, the mode decision process is terminated for the depths $(h+1 \ldots h_{max})$.

2. *early skip detection* (ESD) [23]
   When ESD is enabled, the motion information and *Coded Block Flags* (CBFs) are checked at the earliest possible stage. This is conducted by evaluating $PART\_2N{\times}2N$ before the Skip/Merge modes. If $PART\_2N{\times}2N$ at depth h contains only PBs with $CBF = 0$ for both luma and chroma and without additional motion information, ESD sets $M_h = Skip$ and skips the mode exploration for depths $h \ldots h_{max}$.

3. *CBF fast mode* (CFM) [24]
   CFM considers CBFs of the Square, SMP, and AMP modes at each quadtree depth h. After evaluation at depth h, PBs with $CBF = 0$ are selected as M_h. In this case, the mode decision continues to the depth h + *1* (when available) without further considering the remaining modes at depth h.

4.1 Analysis

After the test set for parametric exploration was identified, the TUT Task Manager (explained in Chapter 3) was used to distribute, run, and gather all the results. Altogether, almost 4000 test runs were required with estimated 10 000 hours (nearly 14 months) of CPU time. Test runs took 2.8 hours to complete on average and resulted in 53 GiB of output data. [1]

4.1.1 Setup

In this work, HM MP (HM 8.0) acts as a base for the optimized mode decision scheme development. In this anchor, the optional early termination mechanisms

[1] Task data is gathered from archived log files that include run time and related info. Output files include video bitstreams and log files.

Table 4.2 *Test Sequences.*

Class	Format	Sequence	Frames	FPS	RA	LB
A	2560×1600 (1600p)	Traffic^{L}	150	30	x	
		PeopleOnStreet^{H}	150	30	x	
B	1920×1080 (1080p)	Kimono	240	24	x	x
		ParkScene^{L}	240	24	x	x
		Cactus	500	50	x	x
		BQTerrace	600	60	x	x
		BasketballDrive^{H}	500	50	x	x
C	832×480 (WVGA)	RaceHorses^{H}	300	30	x	x
		BQMall^{L}	600	60	x	x
		PartyScene	500	50	x	x
		BasketballDrill	500	50	x	x
D	416×240 (WQVGA)	RaceHorses^{H}	300	30	x	x
		BQSquare^{L}	600	60	x	x
		BlowingBubbles	500	50	x	x
		BasketballPass	500	50	x	x
E	1280×720 (720p)	FourPeople	600	60		x
		Johnny^{L}	600	60		x
		KristenAndSara^{H}	500	50		x

Table 4.3 *Profiling Platform.*

Processor	Intel Core 2 Duo E8400 (2×3.0 GHz)
Memory	8 GB
L1 cache	2×32 KB (instruction) + 2×32 KB (data)
L2 cache	6 MB
Compiler	Microsoft Visual C++ 2010
Operating system	Microsoft Windows 7 Enterprise 64-bit

(ECU, ESD, and CFM) are disabled. Testing is done under the RA and LB configurations with the test sequences (classes A-E) listed in Table 4.2 and the $QP \in \{22, 27, 32, 37\}$ from the common test conditions [25]. Sequences identified with x in Table 4.2 are involved in the RA and LB cases. In the previous analysis, the sequences in each class are identified as most and least complex [4] and marked here with ^{H} and ^{L}, respectively.

1. Rate-distortion analysis setup

   The sequence-specific bit rate differences for an identical $PSNR_{AVG}$ values are used as a base for the RD comparisons. $PSNR_{AVG}$ is a weighted average of luma ($PSNR_Y$) and chroma ($PSNR_U$ and $PSNR_V$) PSNR components [7].

The color format for all test sequences (Table 4.2) is YUV 4:2:0, for which values of $PSNR_{AVG}$ are computed per picture as

$$PSNR_{AVG} = (6 \times PSNR_Y + PSNR_U + PSNR_V)/8 \tag{4.1}$$

The final $PSNR_{AVG}$ over the whole sequence is calculated by averaging the obtained picture-specific $PSNR_{AVG}$ values. The sequence-specific bit rate differences have been compared in terms of the *Bjøntegaard delta bit rate* (BD-rate). [26]

Experimentally specified RD points, representing the $QP \in \{22, 27, 32, 37\}$, are used to interpolate an RD curve of a scheme for BD-rate computation. The same set of QP values has also been used to report QP-specific Δ bit rates of the optimized schemes. In practice, these QP-specific Δ bit rates have been obtained by matching the QP-specific $PSNR_{AVG}$ values of the optimized schemes to the RD curve of the default scheme. Sequence-specific results are averaged for the analysis results.

2. Complexity analysis setup

   Table 4.3 lists the specifications for two identical processor platforms used as the profiling environment [4]. SIMD extensions (MMX/SSE/AVX) are disabled and only a single core per processor is used in order to maintain platform independency in the results.

   The *Intel VTune Amplifier XE* profiler is used for reporting an estimated cycle counts for each encoder function. Evaluating internal complexities of these functions with the cycle-level profiling gives a more reliable results than relying only on monitoring function calls. Profiling time was obtained by analyzing only the high and low complexity sequences marked with $^{\mathrm{H}}$ and $^{\mathrm{L}}$ in Table 4.2 and reporting the averages.

4.2 Previous work

Table 4.4 *Impact of ASR, ECU, ESD, and CFM in HM 8.0.*

Reference	Enc.	RA		LB	
		BD-rate	Time	BD-rate	Time
Bossen et al. [27]	HM 8.0	2.3%	-56%	1.3%	-45%

Table 4.5 *RDC evaluations as a function of N_{MAX} and N_{MIN}.*

Reference	Enc.	Condition		RA		LB	
		N_{MAX}	N_{MIN}	BD-rate	Time	BD-rate	Time
Ohm et al. [7]	HM 8.0	16	4	2.2%	-18%	3.7%	-17%
		8	4	11.0%	-42%	17.4%	-42%
Yuan et al. [8]	HM 6.0	16	4	2.2%	-21%	2.9%	-22%
		8	4	12.0%	-44%	14.4%	-43%
		32	8	8.0%	-32%	7.7%	-31%
		32	16	28.2%	-56%	29.9%	-59%

Table 4.6 *RDC evaluations with disabled AMP and SMP.*

Reference	Enc.	Condition		RA		LB	
		SMP	AMP	BD-rate	Time	BD-rate	Time
Ohm et al. [7]	HM 8.0	x	-	0.9%	-13%	1.2%	-12%
Yuan et al. [9]	HM 7.0	x	-	0.8%	-12%	1.1%	-12%
Kim et al. [8]	HM 6.0	x	-	0.7%	-	1.1%	-
		-	-	3.6%	-	4.3%	-

Work done in [4] presents a comprehensive *Rate-Distortion-Complexity* (RDC) comparison between HEVC (HM 6.0) and AVC reference codec (JM 18.0). In this analysis, JM acts as a baseline for the comparisons. The results show that HEVC reduces bit rate by around 37% at a cost of around 40% increment in coding complexity over AVC. The comparison is done with every essential coding tool enabled.

Analysis of the fast built-in algorithms in HM 8.0 is conducted by Bossen et al. [27]. Mode decision algorithms ECU, ESD, and CFM as well as *adaptive search range* (ASR) for the Inter prediction is included in the analyzed methods. Benchmarks in [27] include all these four methods whose combined RDC impact is summarized in Table 4.4.

ECU, ESD, and CFM are also analyzed by Rhee et al. [28] with other algorithms adopted from AVC for fast mode decision. HM 5.0 is used for benchmarking in this work. Their preference is in hierarchical mode decision schemes and the default ECU, ESD, and CFM rather than in mode selections that are based on motion or spatial uniformity.

The RDC characteristics of individual HM coding tools are evaluated by Correa et al. [29]. Bit rate change of $\geq 1.5\%$ and complexity increase of $\geq 5\%$ is defined as significant impact and is used as the selection criteria for the tools. The default

Table 4.7 *Average area shares of the prediction modes with HM.*

Configuration	QP	Skip	Intra	Square	SMP	AMP
RA	22	45%	9%	22%	12%	11%
	27	57%	8%	16%	10%	8%
	32	66%	7%	12%	9%	7%
	37	73%	6%	9%	7%	5%
LB	22	40%	4%	30%	14%	13%
	27	52%	3%	23%	12%	10%
	32	62%	3%	18%	10%	8%
	37	70%	2%	14%	9%	5%

coding structure of HM with $N_{MAX} = 32$ and $N_{MIN} = 4$ is used as an anchor in this tool set testing.

The RDC characteristics of the HEVC encoder are analyzed by Ohm et al. [7] and Kim et al. [8] as a function of N_{MAX} and N_{MIN}. The obtained results from these works are summarized in Table 4.5 and they are here compared with the default configuration of HM. In case of HM 8.0, the setting $N_{MAX} = 16$ increases BD-rate by 3.7% and 2.2% in the LB and the RA cases, respectively. Complexity reduction remains under 20% in these cases thus advocating the use of the fast built-in methods (Table 4.4). Evaluations with HM 6.0 show that increasing N_{MIN} deteriorates BD-rate even more drastically, which makes $N_{MAX} = 32$ and $N_{MIN} = 4$ the recommended settings.

The effects of disabling AMP modes in Inter prediction is also analyzed by Ohm et al. [7] and Kim et al. [8]. Yuan et al. [9] conduct an equivalent analysis with HM 7.0. Table 4.6 gathers the bit rate overheads from these works. Disabling the AMP modes does not decrease encoding time more than 12-13% but the bit rate increases by 0.8-1.2% due to which AMP modes are enabled by default since HM 7.0. In [8], HM 6.0 is analyzed by combining the effect of disabling SMP/AMP modes. The reported bit rate overhead is around 4% but there is no mention about encoding time difference.

This work presents continuation of [7],[8],[9] by examining various mode decision schemes of HEVC Inter prediction in HM MP. The average area shares of prediction modes are listed in Table 4.7. The shares are calculated by multiplying each selection with the block size (in pixels). The test set in Table 4.2 is used in this experiment. Although being only a fraction ($\leq$ 3%) of the encoding complexity,

Skip and Intra modes still account for 43-73% and 55-79% of the PBs under the LB and RA configurations, respectively. Skip mode takes the majority, 40-75%, of the PBs by itself. Its share grows rapidly as a function of QP, i.e., from 40% to 70% and 45% to 73% in the LB and RA cases, respectively.

The remaining shares account for 27-57% of the PBs in the LB case and 21-45% of the PBs in the RA case. They are coded with the Inter prediction modes (Square, SMP, and AMP modes), including also the Merge mode. The encoding time used for these, however, is 60-70% making Inter prediction the most complex operation in HEVC encoding. [4]

The combined share of the SMP and AMP modes is around half of all of Inter-coded PBs (13-27% in the LB case and 12-23% in the RA case). New potential mode decision schemes are then derived utilizing the outcome of the analysis.

4.3 Proposed method for mode decision

The works described in [7],[8],[9] rely on the supposition that the set of searched modes e is limited in one of the following ways:

1. $e \in \{Square\ (uncond.)\}$
2. $e \in \{Square\ (uncond.), SMP\ (uncond.)\}$
3. $e \in \{Square\ (uncond.), SMP\ (uncond.), AMP\ (cond.)\}$

However, only a fraction of all the HEVC Inter prediction modes are covered with these three schemes.

This work develops mode decision schemes yielding the best possible trade-off between encoding complexity and RD performance. Focus is particularly on the schemes whose maximum BD-rate increment is close to 1% compared to HM MP. This means the examination of potential schemes is done under the following conditions:

- As specified by the HEVC MP, block size $N = 4$ does not use the AMP modes.

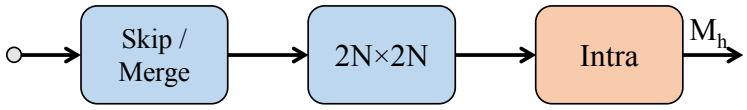


Figure 4.2 *Mode decision without the SMP/AMP modes (S_0), $N \in \{4, 8, 16, 32\}$*

- The evaluation of $PART\_2N\times2N$ is done according to the HM MP specification due to its critical role in the control of HM encoder and in the picture partitioning.
- Bit rate penalty of $> 2\%$ is avoided by limiting the range of N to $N_{MAX} = 32$ and $N_{MIN} = 4$ [7].
- Since contributing $< 3\%$ of the HM MP encoder complexity, Intra and Skip modes are evaluated as specified in HM MP. [4].
- HEVC MP specifies that $PART\_N\times N$ is disabled in Inter coding when $N_{MIN} = 4$.

Justification for focusing on the exploration of the SMP and AMP modes when $N_{SMP} \in \{4, 8, 16, 32\}$ and $N_{AMP} \in \{8, 16, 32\}$ is given by these five conditions. An anchor for the optimizations is HM 8.0, whose default configuration settings are used for the other encoder settings. The optimized schemes are identified as S_x where x represents the sequential number for the schemes.

4.3.1 Mode decision without SMP and AMP modes

Table 4.8 *Impact of disabling AMP and SMP modes on BD-rate.*

Conf.	**ID**	N_{SMP}	N_{AMP}	**Δbit rate**				
				QP 22	**QP 27**	**QP 32**	**QP 37**	**BD-rate**
RA	S_0	∅	∅	3.3%	4.0%	3.9%	3.3%	3.8%
LB	S_0	∅	∅	3.8%	4.7%	4.7%	4.1%	4.5%

Table 4.9 *Impact of disabling AMP and SMP modes on complexity.*

Conf.	**ID**	N_{SMP}	N_{AMP}	**ΔMcpf**				**Avg. ΔMcpf**	**Avg. ΔMcpf$_{enc}$**
				QP 22	**QP 27**	**QP 32**	**QP 37**		
RA	S_0	∅	∅	-68%	-68%	-68%	-67%	-68%	-59%
LB	S_0	∅	∅	-68%	-68%	-68%	-67%	-68%	-61%

Table 4.10 *Impact of unconditional AMP modes on BD-rate.*

Conf.	ID	N_{SMP}	N_{AMP}	Δbit rate QP 22	QP 27	QP 32	QP 37	BD-rate
RA	S_1	{8,16,32,64}	{8,16,32}[a]	-0.3%	-0.4%	-0.6%	-0.7%	-0.5%
LB	S_1	{8,16,32,64}	{8,16,32}[a]	-0.5%	-0.6%	-0.8%	-1.2%	-0.8%

[a] unconditional AMP testing.

Table 4.11 *Impact of unconditional AMP modes on complexity.*

Conf.	ID	N_{SMP}	N_{AMP}	ΔMcpf QP 22	QP 27	QP 32	QP 37	Avg. ΔMcpf	Avg. $\Delta Mcpf_{enc}$
RA	S_1	{8,16,32,64}	{8,16,32}[b]	54%	61%	69%	76%	65%	51%
LB	S_1	{8,16,32,64}	{8,16,32}[b]	57%	65%	72%	80%	69%	56%

[b] unconditional AMP testing.

The impact of SMP and AMP modes for RDC characteristics is examined by configuring $N_{SMP} = \emptyset$ and $N_{AMP} = \emptyset$ in HM encoder, thus disabling them. This highly trimmed and uniform mode decision process for each N (S_0) is depicted in Fig. 4.2.

Tables 4.8 and 4.9 report the RDC figures of S_0 over the default scheme (Fig. 4.1) in the RA and LB cases. Table 4.8 reports the BD-rates and the bit rate differences for four individual QPs ($\Delta bitrate$). The QP-specific complexity differences are given in Table 4.9 as *delta million cycles per frame* ($\Delta Mcpf$), which stands for the combined cycle counts of the mode decision and prediction operations. The prediction operations are included in $\Delta Mcpf$ value since the mode decision process itself has a high impact on the complexity of the prediction modes (Intra, Inter, Skip, and Merge).

The two remaining values characterize the average complexity differences between the compared schemes. The Average $\Delta Mcpf$ value reports an arithmetic mean of the QP-specific $\Delta Mcpf$ values, whereas the Average $\Delta Mcpf_{enc}$ value shows the respective mean at the encoder level by counting all encoding operations.

S_0 increases BD-rate by 3.8% and 4.5% in the RA and LB cases, respectively, but is able to reduce the average prediction complexity by almost 70%. At the encoder level, reducing the complexity by around 60% would make S_0 an desirable solution if the RD penalty would not be an important factor. This limits the usage of S_0 to the cases where complexity is intended to reduce even at the cost of bit rate.

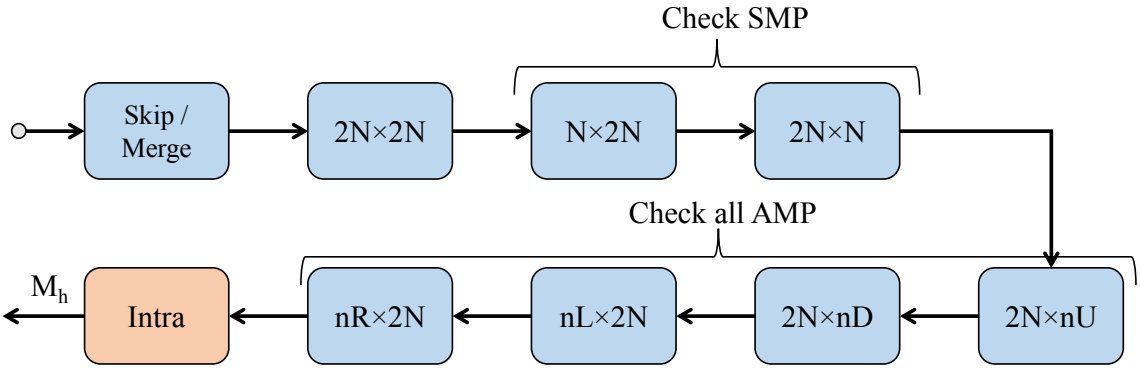


Figure 4.3 *Mode decision with the unconditional AMP modes (S_1), $N \in \{8, 16, 32\}$*

4.3.2 Mode decision with unconditional AMP modes

In this work, the scheme with minimum obtainable complexity is S_0. S_1 is the other corner case. It implements unconditional AMP mode evaluation for $N \in \{8, 16, 32\}$ as visualized in Fig. 4.3. Merge mode only testing can be disabled in this case, since the AMP mode evaluation already contains the functionality. HEVC standard states that the AMP modes are disabled for $N = 4$, so the mode decision with $N = 4$ equals that of Fig. 4.1(b).

Table 4.10 shows that the improvement in BD-rate for S_1 is 0.8% in the LB case and 0.5% in the RA case. However, Table 4.11 reports the complexity overheads for encoding, which are 56% and 51%, respectively. For practical implementations, the cost is considered too high for the RD gain, making the default AMP mode scheme the basis for our optimizations.

4.3.3 Mode decision with limited SMP and AMP depths

Moving from S_0 to S_1 changes the encoding complexity from -61% up to 56% over that of the default configuration. This change is obtained by tuning the coding options for the AMP and AMP mode evaluation. Respective BD-rate changed are from +4.5% to -0.8% in the LB case and from +3.8% to -0.5% in the RA case. This work tests different encoder configurations in order to find the best trade-offs between the S_0 and S_1 corner cases. This has been done as a function of SMP and AMP depths, setting the limit on the range of N_{SMP} and N_{AMP}.

The combined area shares of SMP and AMP modes are tabulated in Table 4.12 over the range of N. Cases when $N \in \{4, 32\}$ are the most potential for exclusion,

Table 4.12 *Combined area shares of SMP and AMP modes with HM.*

Configuration	QP	N=32	N=16	N=8	N=4
RA	22	3%	8%	9%	3%
	27	4%	7%	6%	2%
	32	4%	6%	4%	1%
	37	4%	5%	2%	0%
LB	22	3%	9%	11%	3%
	27	4%	9%	7%	2%
	32	5%	8%	4%	1%
	37	5%	6%	2%	0%

Table 4.13 *Impact of SMP and AMP sizes on BD-rate.*

Conf.	ID	N_{SMP}	N_{AMP}	Δbit rate QP 22	QP 27	QP 32	QP 37	BD-rate
RA	S_2	{8,16,32}	{8,16,32}	1.3%	1.4%	1.0%	0.4%	1.1%
	S_3	{16,32}	{16,32}	2.9%	3.3%	2.8%	1.9%	2.9%
	S_4	{32}	{32}	3.2%	3.9%	3.8%	3.1%	3.7%
	S_5	{4,8,16}	{4,8,16}	0.0%	0.1%	0.2%	0.3%	0.1%
	S_6	{4,8}	{8}	0.4%	0.7%	1.2%	1.5%	0.9%
	S_7	{4}	∅	1.7%	2.2%	2.7%	2.7%	2.4%
LB	S_2	{8,16,32}	{8,16,32}	1.4%	1.6%	1.2%	0.5%	1.3%
	S_3	{16,32}	{16,32}	3.1%	3.7%	3.4%	2.2%	3.3%
	S_4	{32}	{32}	3.7%	4.7%	4.6%	3.7%	4.4%
	S_5	{4,8,16}	{4,8,16}	0.1%	0.2%	0.4%	0.5%	0.3%
	S_6	{4,8}	{8}	0.7%	1.2%	1.7%	2.2%	1.4%
	S_7	{4}	∅	2.1%	2.8%	3.3%	3.5%	3.0%

since the distribution is centered on $N \in \{8, 16\}$. The distribution center also moves towards the larges PBs as a function of QP.

RD and complexity results are listed for different ranges of N_{SMP} and N_{AMP} in Table 4.13 and in Table 4.14. Small AMP/SMP sizes account for only 0-3%, but when the lower bounds of N_{AMP} and N_{SMP} are raised (S_2-S_4), their importance becomes clear. Disabling $N_{SMP} = 4$ in S_2 also has a complexity benefit of around 15%, but at the BD-rate cost of 1.3% in the LB case and 1.1% in the RA case. BD-rate increases at least by almost 3%, when further limiting the lower bound (S_3 and S_4).

When lowering the upper bounds of N_{AMP} and N_{SMP} (S_5-S_7), BD-rate increase is more moderate. When using S_5 (with disabled $N_{SMP} = N_{AMP} = 32$), BD-rate

Table 4.14 *Impact of SMP and AMP sizes on complexity.*

Conf.	ID	N_{SMP}	N_{AMP}	ΔMcpf QP 22	QP 27	QP 32	QP 37	Avg. ΔMcpf	Avg. $\Delta Mcpf_{enc}$
RA	S_2	{8,16,32}	{8,16,32}	-15%	-15%	-15%	-15%	-15%	-15%
	S_3	{16,32}	{16,32}	-38%	-37%	-36%	-36%	-37%	-34%
	S_4	{32}	{32}	-57%	-57%	-56%	-54%	-56%	-49%
	S_5	{4,8,16}	{4,8,16}	-11%	-11%	-12%	-13%	-12%	-10%
	S_6	{4,8}	{8}	-26%	-27%	-27%	-28%	-27%	-21%
	S_7	{4}	∅	-52%	-53%	-57%	-51%	-53%	-45%
LB	S_2	{8,16,32}	{8,16,32}	-14%	-14%	-14%	-19%	-15%	-16%
	S_3	{16,32}	{16,32}	-38%	-37%	-36%	-35%	-36%	-34%
	S_4	{32}	{32}	-58%	-57%	-55%	-54%	-56%	-51%
	S_5	{4,8,16}	{4,8,16}	-11%	-17%	-20%	-13%	-15%	-14%
	S_6	{4,8}	{8}	-27%	-27%	-28%	-28%	-28%	-23%
	S_7	{4}	∅	-54%	-54%	-54%	-53%	-53%	-46%

penalties of 0.3% and 0.1% in the LB case and in the RA case are observed, respectively. S_5 outperforms the overall RDC characteristics of S_2 and the complexity decrease is only 10-14%. Using low QPs with S_5 shows its full effectiveness. Comparing S_6 and S_2 shows the low importance of larger PBs.

4.3.4 SMP/AMP mode only decision

Table 4.15 *Impact of SMP and AMP modes on BD-rate.*

Conf.	ID	N_{SMP}	N_{AMP}	Δbit rate QP 22	QP 27	QP 32	QP 37	BD-rate
RA	S_8	∅	{8,16,32}	1.9%	2.2%	2.0%	1.6%	2.0%
	S_9	{4,8,16,32}	∅	0.8%	0.9%	1.0%	1.1%	0.9%
LB	S_8	∅	{8,16,32}	2.1%	2.6%	2.4%	1.9%	2.4%
	S_9	{4,8,16,32}	∅	0.9%	1.1%	1.4%	1.3%	1.2%

Table 4.16 *Impact of SMP and AMP modes on complexity.*

Conf.	ID	N_{SMP}	N_{AMP}	ΔMcpf QP 22	QP 27	QP 32	QP 37	Avg. ΔMcpf	Avg. $\Delta Mcpf_{enc}$
RA	S_8	∅	{8,16,32}	-47%	-51%	-54%	-57%	-52%	-44%
	S_9	{4,8,16,32}	∅	-17%	-14%	-12%	-9%	-13%	-13%
LB	S_8	∅	{8,16,32}	-49%	-52%	-55%	-58%	-53%	-47%
	S_9	{4,8,16,32}	∅	-17%	-14%	-11%	-8%	-13%	-12%

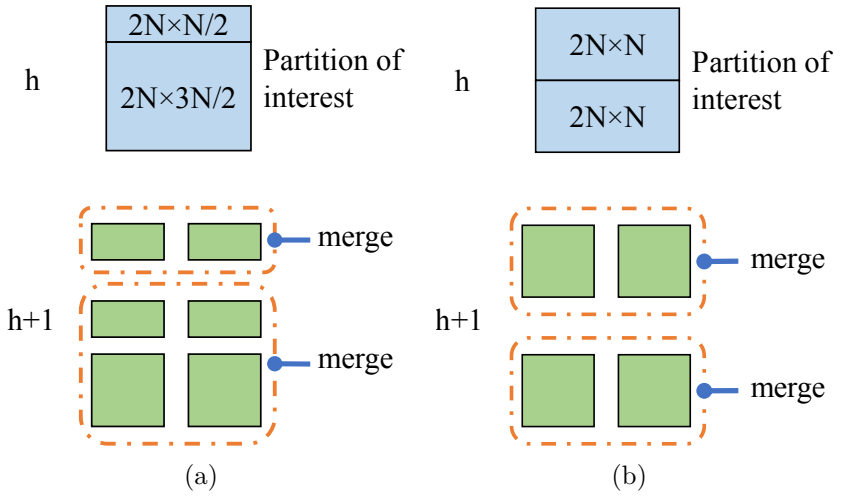


Figure 4.4 *Composing the partition of interest from the smaller blocks with the Merge mode. (a) PART_2N×nU, AMP off. (b) PART_2N×N, SMP off.*

Square and SMP modes are included in HEVC Inter prediction as essential parts in existing approaches [7],[8],[9]. AMP modes are executed either conditionally, optionally, or disabled completely. Fig. 4.1(b) can be adopted as a mode decision scheme without AMP modes, for each N. HM 6.0 and older HM versions used this scheme as the default configuration. Since AMP modes have been shown to improve BD-rate by around 1% with complexity increase of less than 15%, they have been included in the default configurations since HM 7.0 of HM MP (Table 4.6).

SMP modes are given priority in the HM MP default mode decision by unconditionally evaluating them, whereas AMP mode evaluation is limited with predefined conditions (Fig. 4.1). The share of AMP modes is still only 2% smaller despite the priorization of SMP modes (Table 4.7). In addition, the SMP modes are easier to be replaced by other partition modes than the AMP modes.

Merging smaller square (or symmetric, if available) PBs to compensate the disabled AMP modes of interest is illustrated in Fig. 4.4(a). Two Squares and four SMPs at level $h+1$ can replace an AMP block at level h. Respective case for the disabled SMP modes of interest is illustrated in Fig. 4.4(b). Four smaller Squares can be merged to compensate the SMP at level h. In this case, even smaller blocks can be utilized for the prediction result to meet the partition of interest. Coding schemes with the unconditional/disabled SMP modes and the enabled AMP modes are thus worth the consideration.

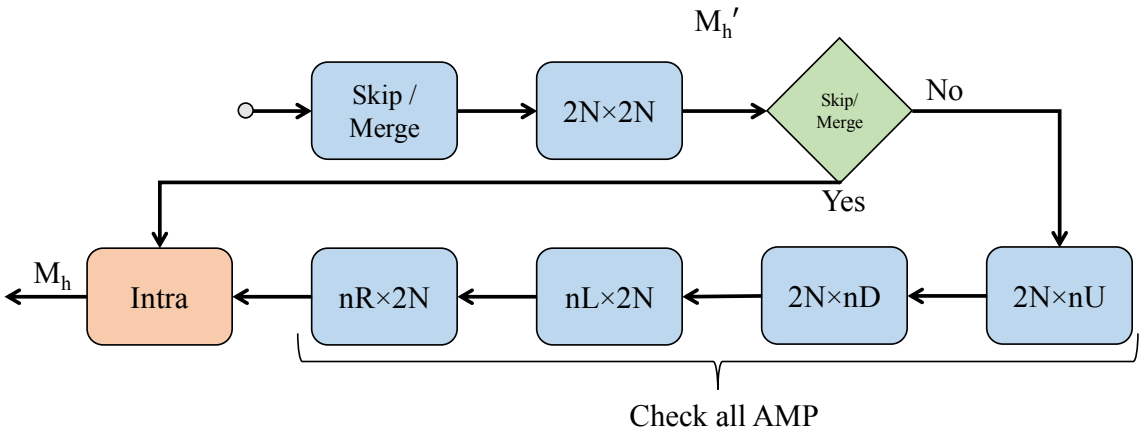


Figure 4.5 *Mode decision with SMP modes disabled (S_8), $N \in \{8, 16\}$.*

Mode decision scheme excluding the SMP modes for $N \in \{8, 16\}$ (S_8) is depicted in Fig. 4.5. Fig. 4.2 illustrates the respective version for $N \in \{4, 32\}$. Table 4.16 tabulates the complexity results for S_8 and for the scheme without the AMP modes (S_9). The respective RD results are given in Table 4.15. The effect of disabling the AMP modes (Table 4.6) is in line with [7].

Disabling the SMP modes increases BD-rate by at least 2%, doubling the penalty of the case when AMP modes are disabled. However, the respective complexity reduction is 47% in the LB case giving a gain factor of almost 4× over that of disabling the AMP modes, although there are twice as many AMP modes to test (Fig. 2.1). Respective complexity decrease is 44% (3.5×) in the RA case. Conditional AMP mode evaluation is the main cause of this complexity inconsistency. Hence, SMP mode evaluation is the more potential for optimizations.

4.3.5 Proposed mode decision for SMP

The first candidate for an optimized mode decision scheme (S_{10}) is depicted in Fig. 4.6. In S_{10}, only Skip, Merge, and Square modes are used for the decision of the best temporary mode (M_h''), and after that the SMP mode evaluation is conditionally done. AMP/SMP modes are skipped if $M_h'' \in \{Skip, Merge\}$ holds, and process continues to Intra mode evaluation. Otherwise, evaluations of the AMP/SMP modes are continued, but the preconditions are not used for evaluating AMP modes as in the default scheme (Fig. 4.1). According to the experiments, this *joint* (j) precondition of SMP and AMP modes is able to reduce complexity by as much as 40% in the LB case and 37% in the RA case. BD-rate increments of this case are 1.7% and

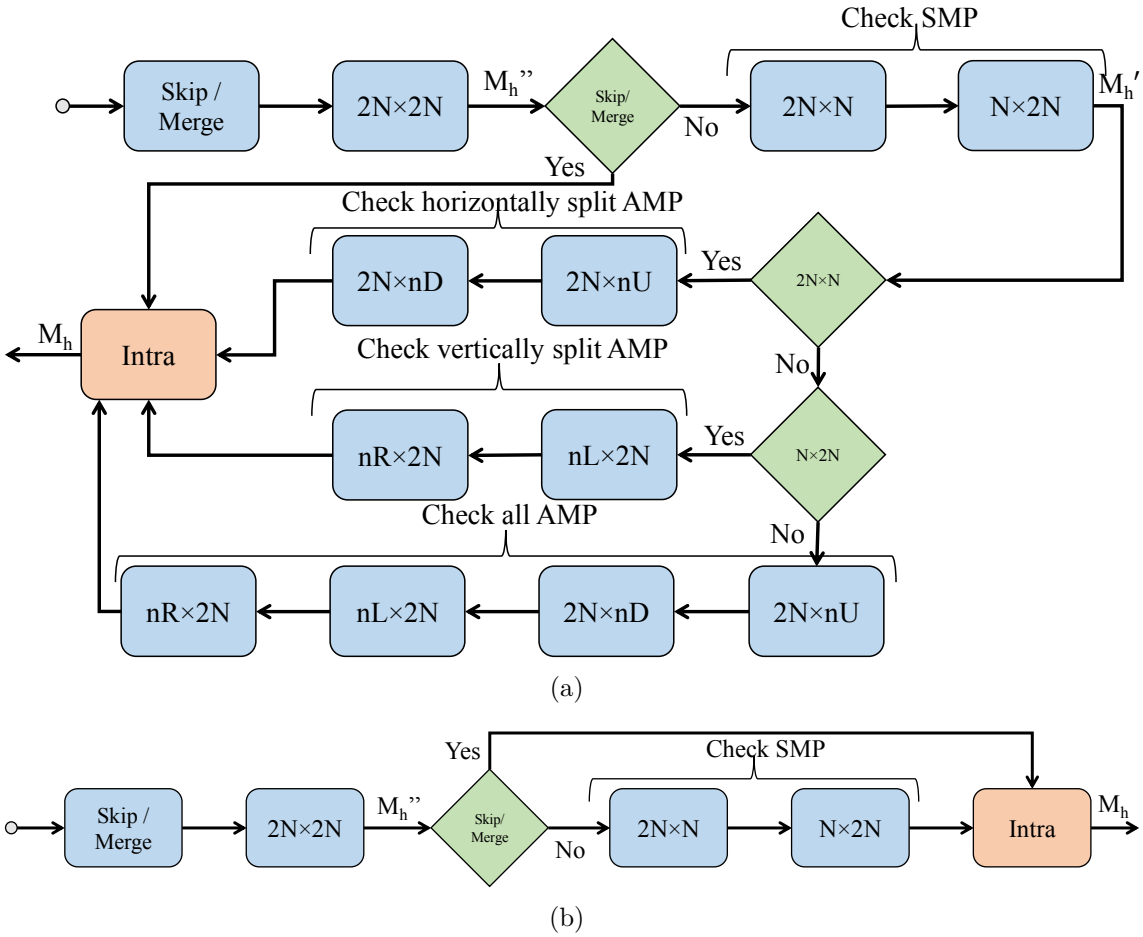


Figure 4.6 *Optimized mode decision with joint (j) precondition for the SMP/AMP modes (S_{10}). (a) $N \in \{8, 16\}$. (b) $N \in \{4, 32\}$.*

1.2% over the default scheme, respectively. This is too large penalty due to which a precondition j is considered a too strict condition for SMP modes.

The second candidate scheme, S_{14}, is presented in Fig. 4.7. It uses a looser precondition for SMP modes lowering the BD-rate loss from that of S_{10}. Only $M_h'' = Skip$ is tested before evaluating the SMP modes and the Merge mode precondition is applied before checking the AMP modes. That is, this scheme only repositions the Skip mode comparison when compared with the default scheme. BD-rate loss was diminished down to 0.3% in the LB case and 0.2% in the RA case when using the distributed (d) preconditions for the SMP and AMP modes. Encoding complexity savings are still 32% and 31% compared with the default scheme, respectively.

RDC characteristics of S_{10}-S_{14} are tabulated in the Tables 4.17 and 4.18. S_{10} is

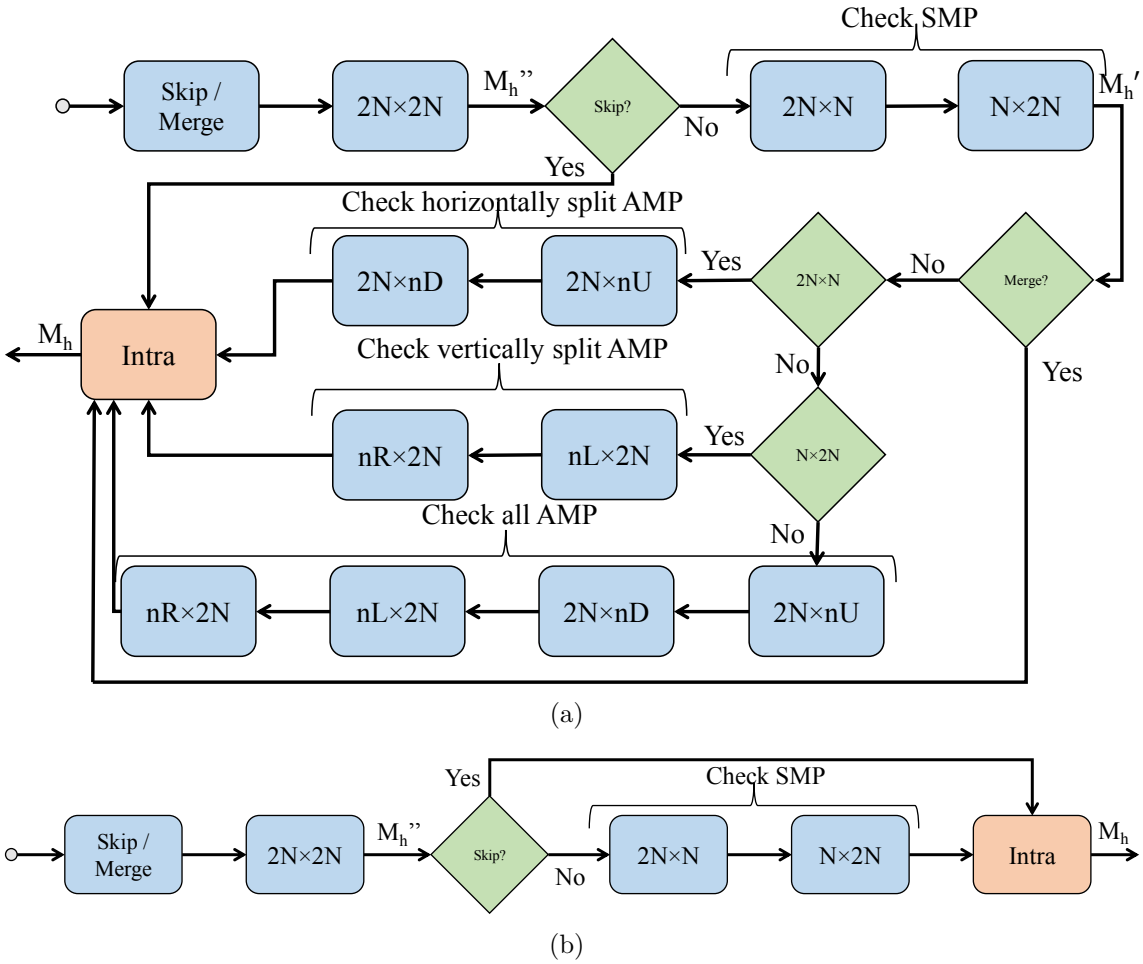


Figure 4.7 *Optimized mode decision with distributed (d) precondition for the SMP/AMP modes (S_{14}). (a) $N \in \{8, 16\}$. (b) $N \in \{4, 32\}$.*

depicted in Fig. 4.6, S_{14} in Fig. 4.7, and the remaining ones are three potential hybrid solutions between them. The impact of shifting the boundary of the preconditions d and j gradually towards the larger N_{SMP} values are benchmarked in S_{11}-S_{13}. As seen in Table 4.13, the relative increase in BD-rate is decreased as a function of N_{SMP}, making S_{11}-S_{13} more preferable compared with other possible hybrid schemes. This also means that the smaller SMP blocks are of higher importance. By gradually adopting the looser precondition d in the increasing order of N_{SMP}, as in S_{11}-S_{13}, the most beneficial RDC trade-offs can be found. This holds even when comparing with the other available hybrid solutions.

The N_{SMP} range limitations of S_{11}, S_{12}, and S_{13} have been adopted to S_{17}, S_{16}, and S_{15}, respectively, by disabling SMP sizes that are dependent on the precondition j. When the block size of interest is in the N_{SMP} range, the functionality of S_{15}-S_{17} is

Table 4.17 *Impact of SMP preconditions on BD-rate.*

Conf.	ID	N_{SMP}	N_{AMP}	Δbit rate QP 22	QP 27	QP 32	QP 37	BD-rate
RA	S_{14}	$\{4^{d},8^{d},16^{d},32^{d}\}$	{8,16,32}	0.1%	0.2%	0.3%	0.3%	0.2%
	S_{13}	$\{4^{d},8^{d},16^{d},32^{j}\}$	{8,16,32}	0.1%	0.2%	0.3%	0.3%	0.2%
	S_{12}	$\{4^{d},8^{d},16^{j},32^{j}\}$	{8,16,32}	0.2%	0.3%	0.5%	0.5%	0.4%
	S_{11}	$\{4^{d},8^{j},16^{j},32^{j}\}$	{8,16,32}	0.4%	0.6%	0.8%	0.8%	0.6%
	S_{10}	$\{4^{j},8^{j},16^{j},32^{j}\}$	{8,16,32}	1.1%	1.3%	1.3%	1.1%	1.2%
LB	S_{14}	$\{4^{d},8^{d},16^{d},32^{d}\}$	{8,16,32}	0.2%	0.3%	0.4%	0.5%	0.3%
	S_{13}	$\{4^{d},8^{d},16^{d},32^{j}\}$	{8,16,32}	0.2%	0.3%	0.5%	0.5%	0.4%
	S_{12}	$\{4^{d},8^{d},16^{j},32^{j}\}$	{8,16,32}	0.3%	0.5%	0.6%	0.8%	0.5%
	S_{11}	$\{4^{d},8^{j},16^{j},32^{j}\}$	{8,16,32}	0.7%	0.8%	1.1%	1.1%	0.9%
	S_{10}	$\{4^{j},8^{j},16^{j},32^{j}\}$	{8,16,32}	1.5%	1.7%	1.7%	1.5%	1.7%

[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).
[j] Disable SMPs if $M''_h \in \{Skip, Merge\}$ (Fig. 4.6).

Table 4.18 *Impact of SMP preconditions on complexity.*

Conf.	ID	N_{SMP}	N_{AMP}	ΔMcpf QP 22	QP 27	QP 32	QP 37	Avg. ΔMcpf	Avg. $\Delta Mcpf_{enc}$
RA	S_{14}	$\{4^{d},8^{d},16^{d},32^{d}\}$	{8,16,32}	-23%	-32%	-39%	-45%	-35%	-31%
	S_{13}	$\{4^{d},8^{d},16^{d},32^{j}\}$	{8,16,32}	-26%	-34%	-41%	-46%	-37%	-32%
	S_{12}	$\{4^{d},8^{d},16^{j},32^{j}\}$	{8,16,32}	-29%	-36%	-43%	-48%	-39%	-34%
	S_{11}	$\{4^{d},8^{j},16^{j},32^{j}\}$	{8,16,32}	-33%	-39%	-44%	-50%	-42%	-36%
	S_{10}	$\{4^{j},8^{j},16^{j},32^{j}\}$	{8,16,32}	-37%	-41%	-45%	-50%	-43%	-37%
LB	S_{14}	$\{4^{d},8^{d},16^{d},32^{d}\}$	{8,16,32}	-23%	-32%	-39%	-45%	-35%	-32%
	S_{13}	$\{4^{d},8^{d},16^{d},32^{j}\}$	{8,16,32}	-26%	-34%	-41%	-47%	-37%	-33%
	S_{12}	$\{4^{d},8^{d},16^{j},32^{j}\}$	{8,16,32}	-30%	-37%	-43%	-49%	-40%	-36%
	S_{11}	$\{4^{d},8^{j},16^{j},32^{j}\}$	{8,16,32}	-35%	-41%	-46%	-51%	-43%	-38%
	S_{10}	$\{4^{j},8^{j},16^{j},32^{j}\}$	{8,16,32}	-39%	-42%	-46%	-51%	-45%	-40%

[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).
[j] Disable SMPs if $M''_h \in \{Skip, Merge\}$ (Fig. 4.6).

equal to that of S_{14} (Fig. 4.7). Otherwise, these schemes as simplified to Fig. 4.2 for $N = 32$ and Fig. 4.5 for $N \in \{8, 16\}$. The schemes S_{15}-S_{17} can be viewed as simplified versions of S_{14}. Compared to S_{10} and hybrid schemes, better trade-offs are obtained with S_{14} and its simplified versions, when looking at the RDC values in Tables 4.19 and 4.20. An average encoding complexity reduction of around 4% and respective BD-rate penalty of under 0.2% is obtained by disabling the precondition *j*, thus limiting N_{SMP} range.

Table 4.19 *Impact of limited SMP range on BD-rate.*

Conf.	ID	N_{SMP}	N_{AMP}	Δbit rate QP 22	QP 27	QP 32	QP 37	BD-rate
RA	S_{14}	{4[d],8[d],16[d],32[d]}	{8,16,32}	0.1%	0.2%	0.3%	0.3%	0.2%
	S_{15}	{4[d],8[d],16[d]}	{8,16,32}	0.1%	0.2%	0.4%	0.4%	0.3%
	S_{16}	{4[d],8[d]}	{8,16,32}	0.2%	0.4%	0.6%	0.8%	0.5%
	S_{17}	{4[d]}	{8,16,32}	0.6%	0.8%	1.1%	1.2%	0.9%
LB	S_{14}	{4[d],8[d],16[d],32[d]}	{8,16,32}	0.2%	0.3%	0.4%	0.5%	0.3%
	S_{15}	{4[d],8[d],16[d]}	{8,16,32}	0.2%	0.3%	0.5%	0.7%	0.4%
	S_{16}	{4[d],8[d]}	{8,16,32}	0.4%	0.6%	0.8%	1.1%	0.7%
	S_{17}	{4[d]}	{8,16,32}	0.8%	1.0%	1.3%	1.4%	1.1%

[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).

Table 4.20 *Impact of limited SMP range on complexity.*

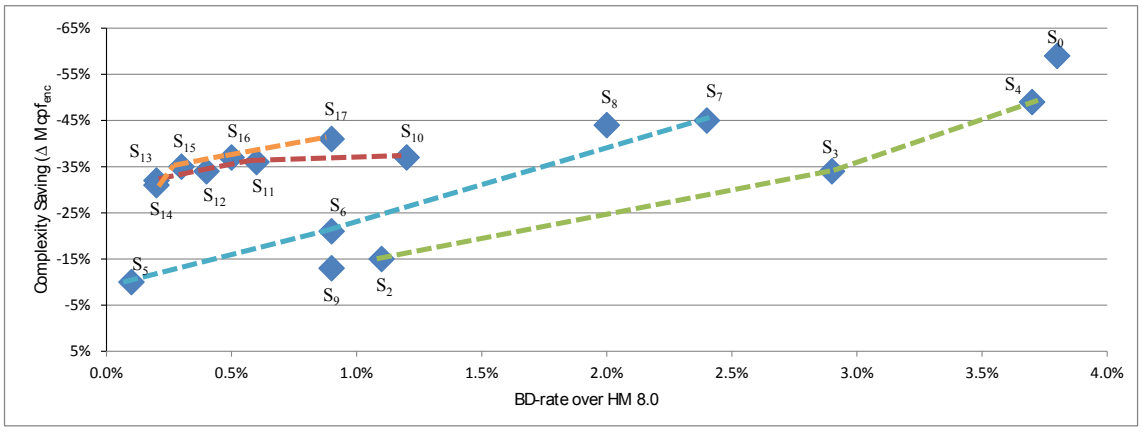
Conf.	ID	N_{SMP}	N_{AMP}	ΔMcpf QP 22	QP 27	QP 32	QP 37	Avg. ΔMcpf	Avg. ΔMcpf_{enc}
RA	S_{14}	{4[d],8[d],16[d],32[d]}	{8,16,32}	-23%	-32%	-39%	-45%	-35%	-31%
	S_{15}	{4[d],8[d],16[d]}	{8,16,32}	-30%	-38%	-45%	-50%	-41%	-35%
	S_{16}	{4[d],8[d]}	{8,16,32}	-35%	-42%	-48%	-53%	-44%	-37%
	S_{17}	{4[d]}	{8,16,32}	-42%	-47%	-52%	-56%	-49%	-41%
LB	S_{14}	{4[d],8[d],16[d],32[d]}	{8,16,32}	-23%	-32%	-39%	-45%	-35%	-32%
	S_{15}	{4[d],8[d],16[d]}	{8,16,32}	-30%	-38%	-45%	-51%	-41%	-36%
	S_{16}	{4[d],8[d]}	{8,16,32}	-36%	-43%	-49%	-54%	-45%	-40%
	S_{17}	{4[d]}	{8,16,32}	-44%	-48%	-53%	-57%	-50%	-44%

[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).

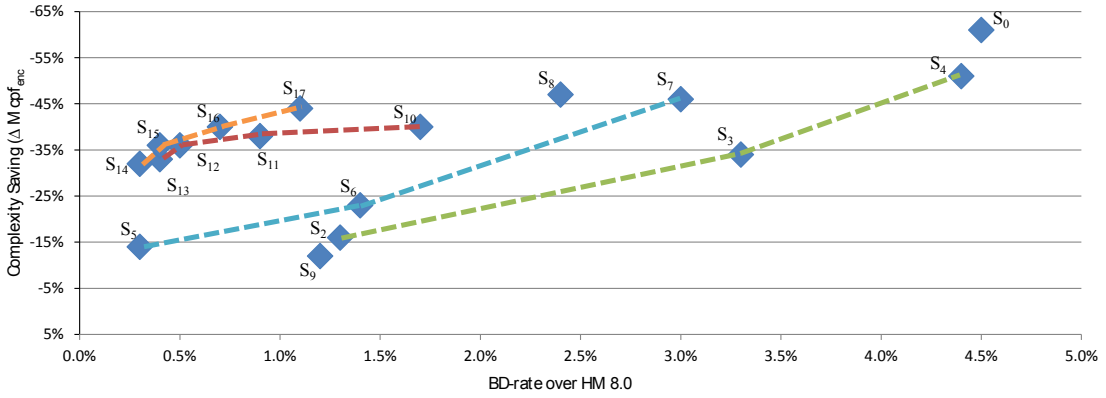
Fig. 4.8 graphically illustrates RDC characteristics of S_{14}-S_{17} over those of the previously introduced decision schemes. Scheme S_1 is left outside of the plot due to its complexity increase over HM MP. The schemes are grouped (S_2-S_4, S_5-S_7, S_{10}-S_{13}, and S_{14}-S_{17}) by connecting them with dashed lines to improve their visual comparison. Among the scheme groups, a common trend is seen between the LB and RA cases. The best RDC trade-off is achieved with S_{14}-S_{17} and their N_{SMP} ranges are used as the baseline for AMP mode evaluation.

4.3.6 Proposed AMP mode decision

Selecting the best possible preconditions for the SMP modes and N_{SMP} ranges for them is highly important as shown by the experiments. For AMP modes,



(a)



(b)

Figure 4.8 *Comparison of the optimized mode decision schemes by RDC characteristics. (a) the RA case. (b) the LB case.*

preconditions are enabled by default in HM MP, so the default AMP range, i.e., $N_{AMP} \in \{8, 16, 32\}$, is optimized in this work.

RDC values for eight candidate configurations (S_{18}-S_{25}) are tabulated in Tables 4.21 and 4.22. These configurations combine limited N_{AMP} ranges with the best four N_{SMP} ranges (Table 4.19). Changing $N_{AMP} \in \{8, 16, 32\}$ to $N_{AMP} \in \{8, 16\}$ in S_{18}-S_{21} means disabling the Merge mode only testing (Table 4.1) for $N_{AMP} = 32$. This optimization gives encoding complexity saving of over 2% with almost no BD-rate overhead in the RA case. The LB case has respective values of 2% and 0.1%. Much higher impact on the RDC characteristics is achieved by limiting $N_{AMP} \in \{8, 16\}$ to $N_{AMP} = 8$ in S_{22}-S_{25}. This way, the encoding complexity is further reduced by additional 6-8% with extra BD-rate overhead of 0.5-0.8% in the LB case. The

Table 4.21 *Impact of limited AMP range on BD-rate.*

Conf.	ID	N_{SMP}	N_{AMP}	Δbit rate QP 22	QP 27	QP 32	QP 37	BD-rate
RA	S_{18}	{4[d],8[d],16[d],32[d]}	{8,16}	0.1%	0.2%	0.4%	0.4%	0.3%
	S_{19}	{4[d],8[d],16[d]}	{8,16}	0.1%	0.3%	0.4%	0.5%	0.3%
	S_{20}	{4[d],8[d]}	{8,16}	0.2%	0.4%	0.7%	0.9%	0.5%
	S_{21}	{4[d]}	{8,16}	0.6%	0.8%	1.1%	1.3%	0.9%
	S_{22}	{4[d],8[d],16[d],32[d]}	{8}	0.3%	0.5%	0.7%	0.8%	0.6%
	S_{23}	{4[d],8[d],16[d]}	{8}	0.3%	0.6%	0.9%	1.0%	0.7%
	S_{24}	{4[d],8[d]}	{8}	0.5%	0.9%	1.3%	1.6%	1.1%
	S_{25}	{4[d]}	{8}	0.9%	1.3%	1.8%	2.1%	1.5%
LB	S_{18}	{4[d],8[d],16[d],32[d]}	{8,16}	0.2%	0.3%	0.5%	0.5%	0.4%
	S_{19}	{4[d],8[d],16[d]}	{8,16}	0.2%	0.4%	0.7%	0.9%	0.6%
	S_{20}	{4[d],8[d]}	{8,16}	0.4%	0.7%	1.0%	1.3%	0.8%
	S_{21}	{4[d]}	{8,16}	0.8%	1.1%	1.4%	1.6%	1.2%
	S_{22}	{4[d],8[d],16[d],32[d]}	{8}	0.5%	0.8%	1.1%	1.3%	0.9%
	S_{23}	{4[d],8[d],16[d]}	{8}	0.5%	0.9%	1.3%	1.7%	1.1%
	S_{24}	{4[d],8[d]}	{8}	0.8%	1.4%	1.8%	2.4%	1.6%
	S_{25}	{4[d]}	{8}	1.3%	1.8%	2.4%	2.7%	2.0%

[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).

respective values in the RA case are 5-8% and 0.3-0.6%.

Impact of the limited N_{AMP} range is visualized in Fig. 4.9 by tracing the RDC characteristics of S_{14}-S_{17}, S_{18}-S_{21}, and S_{22}-S_{25} with a dashed line. It has been zoomed in from Fig. 4.8 for better illustration of closely distributed RDC value differences. The top RDC characteristics are obtained with these scheme groups. However, none of them provides the best gain alone as indicated by the crossing lines. Tables 4.19, 4.20, 4.21, and 4.22 tabulate the QP-specific values, which show a possibility for using the variable ranges of N_{SMP} and N_{AMP} to compose a single scheme with improved RDC characteristics.

4.3.7 Proposed QP-specific mode decision

Tables 4.23, 4.24, 4.25, and 4.26 gather the most favored QP-specific ranges of N_{SMP} and N_{AMP} from Tables 4.19, 4.20, 4.21, and 4.22. The step size of five is maintained for the QPs, i.e. $QP \in \{22, 27, 32, 37\}$. To gain more accurate QP-specific specifications, the step size could be shortened. However, this would add to the complexity of the parametrization process of N_{SMP} and N_{AMP}.

Table 4.22 *Impact of limited AMP range on complexity.*

Conf.	ID	N_{SMP}	N_{AMP}	ΔMcpf QP 22	ΔMcpf QP 27	ΔMcpf QP 32	ΔMcpf QP 37	Avg. ΔMcpf	Avg. $\Delta Mcpf_{enc}$
RA	S_{18}	{4[d],8[d],16[d],32[d]}	{8,16}	-24%	-33%	-40%	-46%	-36%	-33%
	S_{19}	{4[d],8[d],16[d]}	{8,16}	-32%	-40%	-46%	-51%	-42%	-38%
	S_{20}	{4[d],8[d]}	{8,16}	-37%	-44%	-49%	-54%	-46%	-40%
	S_{21}	{4[d]}	{8,16}	-43%	-48%	-53%	-57%	-50%	-43%
	S_{22}	{4[d],8[d],16[d],32[d]}	{8}	-33%	-40%	-51%	-51%	-44%	-38%
	S_{23}	{4[d],8[d],16[d]}	{8}	-40%	-47%	-52%	-56%	-49%	-43%
	S_{24}	{4[d],8[d]}	{8}	-47%	-53%	-57%	-60%	-54%	-48%
	S_{25}	{4[d]}	{8}	-53%	-57%	-61%	-62%	-58%	-51%
LB	S_{18}	{4[d],8[d],16[d],32[d]}	{8,16}	-24%	-32%	-40%	-46%	-35%	-33%
	S_{19}	{4[d],8[d],16[d]}	{8,16}	-31%	-40%	-46%	-52%	-42%	-39%
	S_{20}	{4[d],8[d]}	{8,16}	-37%	-44%	-50%	-55%	-46%	-42%
	S_{21}	{4[d]}	{8,16}	-45%	-49%	-54%	-57%	-51%	-46%
	S_{22}	{4[d],8[d],16[d],32[d]}	{8}	-32%	-39%	-46%	-50%	-42%	-39%
	S_{23}	{4[d],8[d],16[d]}	{8}	-39%	-47%	-52%	-56%	-49%	-45%
	S_{24}	{4[d],8[d]}	{8}	-47%	-53%	-57%	-60%	-54%	-49%
	S_{25}	{4[d]}	{8}	-57%	-58%	-61%	-63%	-60%	-54%

[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).

The recommendations are culminated in the listing of the best five ranges of N_{SMP} and N_{AMP} per a tested QP. This shortlist is based on the RDC characteristics. The highest prediction complexity drop at a certain Δbit rate value, i.e., the local maximum points, are adopted for each QP of interest. If the number of local maximum points exceeds five, the following RDC conditions are used to select the best ones: for each 0.1% Δbit rate increase, the prediction complexity must decrease at least 2%. In the LB case, the ranges of N_{SMP} and N_{AMP} are adopted from S_{19}, S_{20}, S_{23}, S_{24}, S_{25} for QP = 22, S_{15}, S_{16}, S_{23}, S_{24}, S_{25} for QP = 27, and S_{14}, S_{15}, S_{16}, S_{17}, S_{24} for $QP \in \{32, 37\}$. In the RA case, the respective source schemes are S_{19}, S_{20}, S_{23}, S_{24}, S_{25} for $QP \in \{22, 27, 32\}$ and S_{14}, S_{15}, S_{16}, S_{23}, S_{24} for QP = 37.

Table 4.23 *Shortlist of recommended QP-Specific Ranges of N_{SMP} and N_{AMP}, BD-rate for the RA case.*

N_{SMP} / N_{AMP} QP 22	QP 27	QP 32	QP 37	Δbit rate QP 22	QP 27	QP 32	QP 37
{4[d],8[d],16[d]} {8,16}			{4[d],8[d],16[d],32[d]} {8,16,32}	0.1%	0.3%	0.4%	0.3%
{4[d],8[d]} {8,16}			{4[d],8[d],16[d]} {8,16,32}	0.2%	0.4%	0.7%	0.4%
{4[d],8[d],16[d]} {8}			{4[d],8[d]} {8,16,32}	0.3%	0.6%	0.9%	0.8%
{4[d],8[d]} {8}			{4[d],8[d],16[d]}	0.5%	0.9%	1.3%	1.0%
{4[d]} {8}			{4[d],8[d]}	0.9%	1.3%	1.8%	1.6%

[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).

Table 4.24 *Shortlist of recommended QP-Specific Ranges of N_{SMP} and N_{AMP}, BD-rate for the LB case.*

N_{SMP} / N_{AMP} QP 22	QP 27	QP 32	QP 37	Δbit rate QP 22	QP 27	QP 32	QP 37
{4[d],8[d],16[d]} {8,16}			{4[d],8[d],16[d],32[d]} {8,16,32}	0.2%	0.3%	0.4%	0.5%
{4[d],8[d]} {8,16}			{4[d],8[d],16[d]} {8,16,32}	0.4%	0.6%	0.5%	0.7%
{4[d],8[d],16[d]} {8}			{4[d],8[d]} {8,16,32}	0.5%	0.9%	0.8%	1.1%
{4[d],8[d]} {8}			{4[d]} {8,16,32}	0.8%	1.4%	1.3%	1.4%
{4[d]} {8}			{4[d],8[d]}	1.3%	1.8%	1.8%	2.4%

[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).

Table 4.25 *Shortlist of recommended QP-Specific Ranges of N_{SMP} and N_{AMP}, complexity for the RA case.*

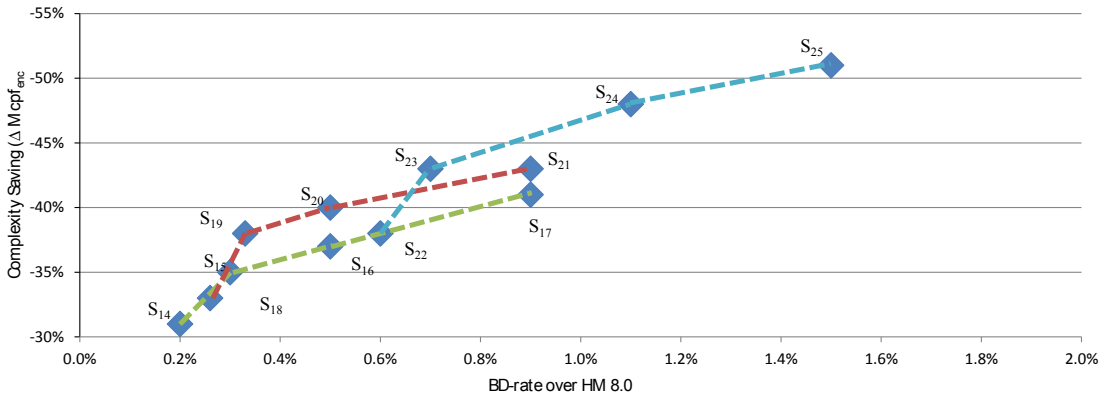
N_{SMP} / N_{AMP}				**ΔMcpf**			
QP 22	**QP 27**	**QP 32**	**QP 37**	**QP 22**	**QP 27**	**QP 32**	**QP 37**
{4[d],8[d],16[d]} {8,16}			{4[d],8[d],16[d],32[d]} {8,16,32}	-32%	-40%	-46%	-45%
{4[d],8[d]} {8,16}			{4[d],8[d],16[d]} {8,16,32}	-37%	-44%	-49%	-50%
{4[d],8[d],16[d]} {8}			{4[d],8[d]} {8,16,32}	-40%	-47%	-52%	-53%
{4[d],8[d]} {8}			{4[d],8[d],16[d]}	-47%	-53%	-57%	-56%
{4[d]} {8}			{4[d],8[d]}	-53%	-57%	-61%	-60%

[d] Disable SMPs if $M_h'' = Skip$ (Fig. 4.7).

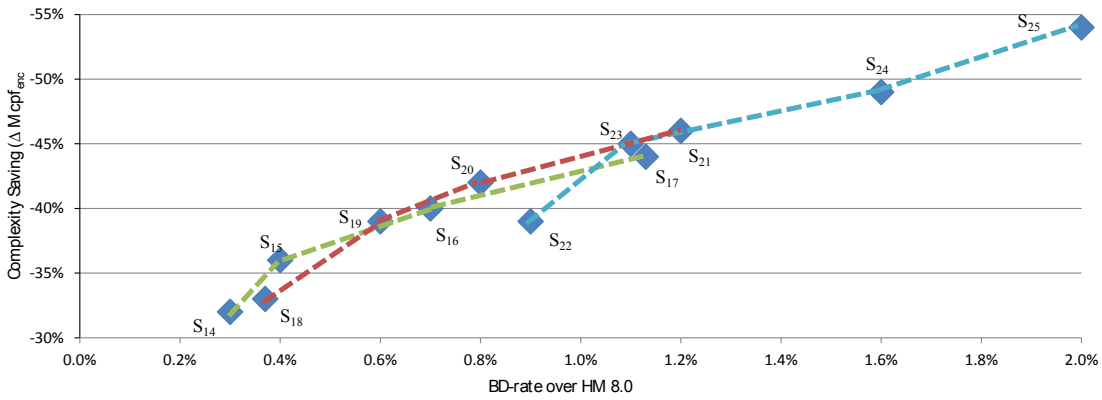
Table 4.26 *Shortlist of recommended QP-Specific Ranges of N_{SMP} and N_{AMP}, complexity for the LB case.*

N_{SMP} / N_{AMP}				**ΔMcpf**			
QP 22	**QP 27**	**QP 32**	**QP 37**	**QP 22**	**QP 27**	**QP 32**	**QP 37**
{4[d],8[d],16[d]} {8,16}			{4[d],8[d],16[d],32[d]} {8,16,32}	-31%	-38%	-39%	-45%
{4[d],8[d]} {8,16}			{4[d],8[d],16[d]} {8,16,32}	-37%	-43%	-45%	-51%
{4[d],8[d],16[d]} {8}			{4[d],8[d]} {8,16,32}	-39%	-47%	-49%	-54%
{4[d],8[d]} {8}			{4[d]} {8,16,32}	-47%	-53%	-53%	-57%
{4[d]} {8}			{4[d],8[d]}	-57%	-58%	-57%	-60%

[d] Disable SMPs if $M_h'' = Skip$ (Fig. 4.7).



(a)



(b)

Figure 4.9 *Comparison of most potential QP-independent mode decision schemes by RDC characteristics. (a) the RA case. (b) the LB case.*

5. GUIDELINES AND COMPARISON

Guidelines for HEVC encoder optimization can be generalized from the analysis results. These optimization guidelines are separately given for QP-specific and QP-independent approaches.

5.1 QP-independent optimization guidelines

When the baseline is the default configuration of HM MP, the following optimization order provides the best operating points in the QP-independent optimization:

1. The Skip mode testing is to be used as a precondition for the SMP modes, that is, set $N_{SMP} \in \{4^{\mathrm{d}}, 8^{\mathrm{d}}, 16^{\mathrm{d}}, 32^{\mathrm{d}}\}$
2. The SMP modes with $N = 32$ are to be disabled, that is, set $N_{SMP} \in \{4^{\mathrm{d}}, 8^{\mathrm{d}}, 16^{\mathrm{d}}\}$
3. The SMP modes with $N = 16$ are to be disabled, that is, set $N_{SMP} \in \{4^{\mathrm{d}}, 8^{\mathrm{d}}\}$
4. The SMP modes with $N = 8$ are to be disabled, that is, set $N_{SMP} \in \{4^{\mathrm{d}}\}$

With this four-step optimization, the complexity of HM MP encoding is decreased gradually up to 44% in the LB case and 41% in the RA case. BD-rate increments are 1.1% and 0.9%, respectively. When the AMP modes are completely disabled (Table 4.15), an almost identical BD-rate penalty is paid. In that case, only one third of the complexity saving is obtained compared to that of this four-step proposal. Thus, the SMP modes involve much higher optimization potential.

The RDC characteristics of these four steps are summarized in Table 5.1. In fact, these are equal to S_{14}-S_{17} recommended after the SMP analysis. The first step is the most significant. It alone reduces the *average* (avg) encoding complexity by over

Table 5.1 *The Proposed QP-Specific Ranges of N_{SMP} and N_{AMP}.*

Cfg.	ID	N_{SMP} / N_{AMP} QP 22	QP 27	QP 32	QP 37	Δbit rate min	max	BD-rate	Δ $Mcpf_{enc}$ min	max	avg
RA	S_{14}	{4[d],8[d],16[d],32[d]} / {8,16,32}				0.1%	0.3%	0.2%	-19%	-42%	-31%
	S_{15}	{4[d],8[d],16[d]} / {8,16,32}				0.1%	0.4%	0.3%	-22%	-45%	-35%
	S_{16}	{4[d],8[d]} / {8,16,32}				0.2%	0.8%	0.5%	-26%	-47%	-37%
	S_{17}	{4[d]} / {8,16,32}				0.6%	1.2%	0.9%	-31%	-50%	-41%
LB	S_{14}	{4[d],8[d],16[d],32[d]} / {8,16,32}				0.2%	0.5%	0.3%	-20%	-43%	-32%
	S_{15}	{4[d],8[d],16[d]} / {8,16,32}				0.2%	0.7%	0.4%	-24%	-47%	-36%
	S_{16}	{4[d],8[d]} / {8,16,32}				0.4%	1.1%	0.7%	-28%	-50%	-40%
	S_{17}	{4[d]} / {8,16,32}				0.8%	1.4%	1.1%	-34%	-52%	-44%

[d] Disable SMPs if $M_h'' = Skip$ (Fig. 4.7).

30%. Complexity decrease of 2-4% per step is achieved after that, with respective BD-rate overhead of 0.1% - 0.4% per each step. The ranges of encoding complexities and Δ bit rates are reported in Table 5.1. In this case, the QP-specific *minimum* (min) and *maximum* (max) bounds are found with QPs of 22 and 37, respectively. Tables 4.23, 4.24, 4.25, and 4.26 contain the QP-specific prediction complexities and Δ bit rates for these steps.

5.2 Guidelines for QP-specific optimizations

Further improvements for the RDC values of the QP-independent four-step proposal can be obtained by QP-specific parameterization of N_{SMP} and N_{AMP}. The same N_{SMP} ranges are reused in this fine-grained parameterization by combining them with a default or a limited range of N_{AMP} and assigning them separately for each QP. Tables 4.23, 4.24, 4.25, and 4.26 list the N_{SMP} and N_{AMP} ranges of the most eligible combinations.

Three QP-specific configuration examples (S_{26}-S_{28}) are listed in Tables 5.2 and 5.3

Table 5.2 *The Proposed QP-Specific Ranges of N_{SMP} and N_{AMP} Listing, BD-rate.*

Cfg.	ID	N_{SMP} / N_{AMP} QP 22	QP 27	QP 32	QP 37	Δbit rate min	max	BD-rate
RA	S_{26}	$\{4^d,8^d\}$ / $\{8\}$		$\{4^d,8^d,16^d\}$ / $\{8,16\}$	$\{8,16,32\}$	0.4%	0.6%	0.5%
	S_{27}	$\{4^d\}$ / $\{8\}$	$\{4^d,8^d\}$	$\{4^d,8^d,16^d\}$		0.9%	1.0%	0.9%
	S_{28}	$\{4^d\}$ / $\{8\}$		$\{4^d,8^d\}$		0.9%	1.6%	1.3%
LB	S_{26}	$\{4^d,8^d,16^d\}$ / $\{8\}$	$\{4^d,8^d\}$ / $\{8,16,32\}$	$\{4^d,8^d,16^d\}$	$\{4^d,8^d,16^d,32^d\}$	0.5%	0.6%	0.5%
	S_{27}	$\{4^d,8^d\}$ / $\{8\}$	$\{4^d,8^d,16^d\}$	$\{4^d,8^d\}$ / $\{8,16,32\}$		0.8%	1.1%	0.9%
	S_{28}	$\{4^d\}$ / $\{8\}$	$\{4^d,8^d\}$	$\{4^d\}$ / $\{8,16,32\}$		1.3%	1.4%	1.3%

[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).

with BD-rate overheads of 0.5%, 0.9%, and 1.3% over that of HM MP, respectively. In all examined schemes, the QP-specific Δ bit rates increase as a function of QP, so QPs between the tested ones are parameterized by the higher one of the tested QPs. This means that the tabulated QP-specific parameterizations of S_{26}-S_{28} apply as a recommendation for $QP = 22$, $QP \in [23, 27]$, $QP \in [28, 32]$, and $QP \in [33, 37]$, respectively. Hence, the given BD-rate limits are safely obtained. Complexity increase of this approach is around 2% on average, but without any BD-rate penalty. CTU-level reparameterization with S_{26}-S_{28} would ease their adaptation to the different QPs, but it has not been implemented in this work.

The additional flexibility offered by the QP-specific schemes makes the desired RDC characteristics more easy to achieve. For example, the complexity drop of S_{28} is 5% over that of S_{17} in the LB case and 9% in the RA case. Additional BD-rate overheads are 0.2% and 0.4%, respectively, bringing it close to that of one step in the four-step proposal with the QP-independent schemes. The gap between the min and max QP-specific values is also reduced with these schemes.

One of the main design principles when developing these optimization techniques has been easy configurability, in addition to the RDC characteristics. Existing control structures of the HEVC encoders can be easily modified to incorporate the precondition d (Fig. 4.7) and the QP-specific parameterization of N_{SMP} and N_{AMP}.

Table 5.3 *The Proposed QP-Specific Ranges of N_{SMP} and N_{AMP} Listing,complexity.*

Cfg.	ID	N_{SMP} / N_{AMP} QP 22	QP 27	QP 32	QP 37	$\Delta\ Mcpf_{enc}$ min	max	avg
RA	S_{26}	$\{4^d,8^d\}$ | $\{8\}$		$\{4^d,8^d,16^d\}$ | $\{8,16\}$	| $\{8,16,32\}$	-38%	-45%	-42%
	S_{27}	$\{4^d\}$ | $\{8\}$	$\{4^d,8^d\}$	$\{4^d,8^d,16^d\}$		-43%	-52%	-47%
	S_{28}	$\{4^d\}$ | $\{8\}$		$\{4^d,8^d\}$		-43%	-55%	-50%
LB	S_{26}	$\{4^d,8^d,16^d\}$ | $\{8\}$	$\{4^d,8^d\}$	$\{4^d,8^d,16^d\}$ | $\{8,16,32\}$	$\{4^d,8^d,16^d,32^d\}$	-34%	-43%	-39%
	S_{27}	$\{4^d,8^d\}$ | $\{8\}$	$\{4^d,8^d,16^d\}$	$\{4^d,8^d\}$ | $\{8,16,32\}$		-40%	-50%	-44%
	S_{28}	$\{4^d\}$ | $\{8\}$	$\{4^d,8^d\}$	$\{4^d\}$ | $\{8,16,32\}$		-47%	-52%	-49%

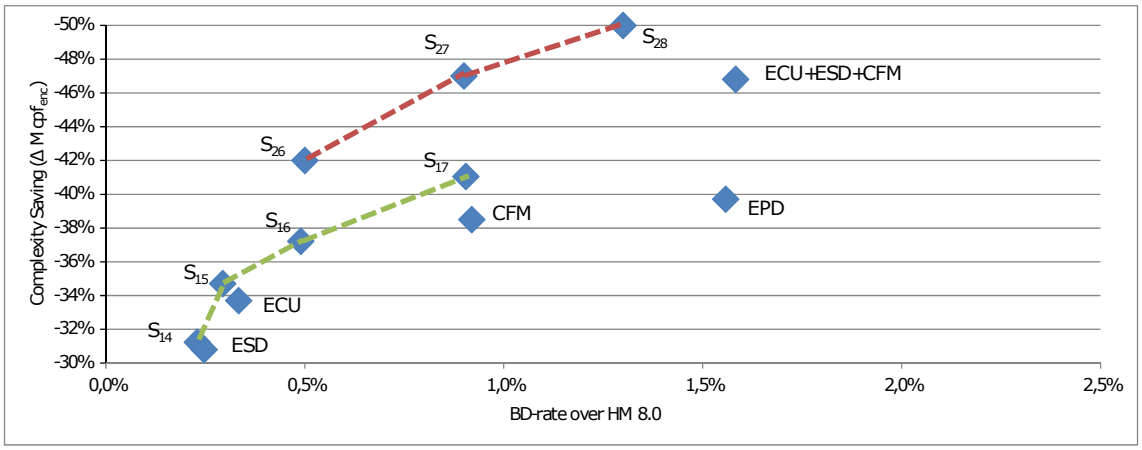
[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).

The schemes do not pose any difficulties in processing adjacent coding tree depths of a single CTU or different CTUs in parallel. Although the conditional SMP/AMP evaluations are effective optimizations when the PBs at each coding tree depth h are processed sequentially, parallel PB processing may easily cancel these benefits. However, parallel processing at level h can be made simpler by limiting SMP/AMP ranges, which effectively reduces the number of PB types to support.

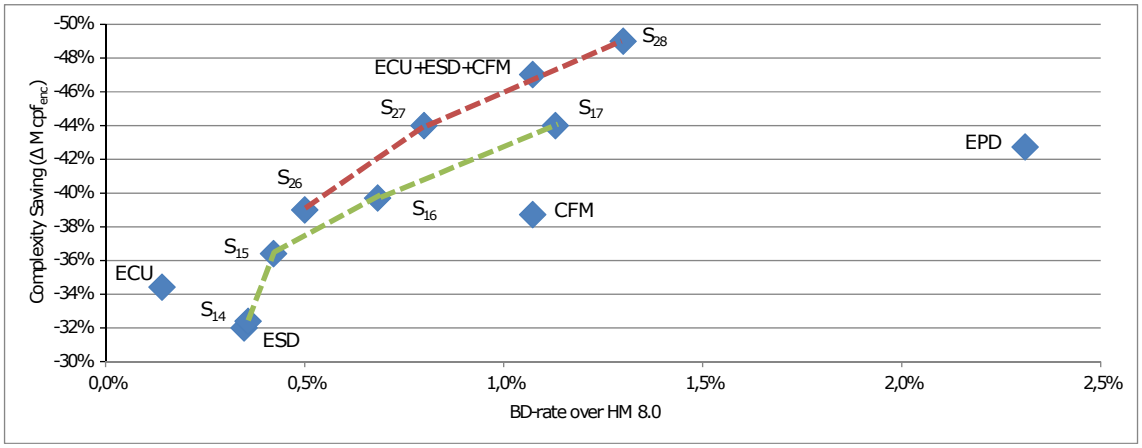
5.3 Comparison with existing techniques

The existing speed-up methods of HM including ECU, ESD, and CFM, are individually benchmarked in [22],[23],[24]. However, all these evaluations use an obsolete version of HM (HM 5.0 or earlier). Only several speed-up method combinations have been tested with HM 8.0 (Table 4.4). Furthermore, only a limited subset of test sequences are used to derivate these results.

In addition, they use the program runtimes for the complexity measurements. ECU, ESD, and CFM are benchmarked in the presented experiments with HM 8.0. To ensure fair comparison, test conditions and the test sets are equal for each scheme. RDC characteristics of ECU, ESD, and CFM are visualized in Fig. 5.1 against S_{14}-S_{17}, S_{26}-S_{28}, the proposed QP-independent schemes, and QP-specific schemes



(a)



(b)

Figure 5.1 *Comparison of most potential (proposed and contemporary) QP-specific mode decision schemes by RDC characteristics. (a) the RA case. (b) the LB case.*

(Tables 5.1, 5.2, and 5.3). In the LB case, ESD and CFM are outperformed by the proposed scheme. ECU, on the other hand, can provide the best RDC trade-off among the schemes with up to 34% complexity drop. If BD-rate overhead of 1.1% is allowed, ECU+ESD+CFM competes with S_{27} and S_{28}. However, ECU+ESD+CFM complexity gap over S_{27} or S_{28} is much wider, because the QP-specific complexity drop varies between -29% and -68% (Tables 5.2 and 5.3).

In the RA case, the build-in methods are outperformed by the proposed schemes, without trade-offs. When comparing ESD, ECU, and CFM to S_{14}, S_{15}, and S_{17}, respectively, the proposed schemes achieve smaller RD overhead and higher complexity reduction. Our optimization techniques also obtain better RDC character-

istics than all the built-in methods together (ECU+ESD+CFM). For example, S_{27} almost halves the BD-rate overhead from 1.6% to 0.9% while achieving an equal complexity loss with ECU+ESD+CFM. In any case, synergy gains between these built-in methods to optimize the HEVC mode decision onward is a potential future work.

The other speed-up techniques introduced to HEVC mode decision [30], [31], [32] use early termination and/or a coding tree depth reduction based on different threshold values. The first approach [30] evaluates probability functions with a Bayesian decision theory. It monitors quadtree partitioning at each quadtree depth h and decides if further split is done. The second approach [31] preselects the depth range of the coding tree as a function of the spatially and temporally related CUs. It also speeds up the pre-limited mode decision by exploiting content-adaptive thresholds. The third approach [32] introduces an algorithm called *early partition decision* (EPD) for terminating the depth search process. EPD uses the average RD cost of a previously coded skipped CUs of the same size as a condition for termination. Square, SMP, and AMP modes are evaluated and if the termination condition holds, the search at depth h and further depths ($h+1...h_{max}$) is terminated. ECU is combined with EPD to boost its RDC values.

RDC results of these three prior-art works are not directly comparable with the results of this work, since HM 4.0 or earlier versions were used as the base. In general, all of them reduce encoding time of HM by around 40%. First approach [30] suffers BD-rate penalty close to 2%. The second approach [31] is assumed to suffer a similar overhead, if the associated PSNR deviation is added to the bit rate differences. The third approach [32] is the most potential one of these works, having BD-rate increase of only around 1% over that of HM 4.0. Fig. 5.1 shows that the proposed approach also outperforms the EPD in the LB and the RA cases. In addition, EPD is considered less hardware friendly, having a complex control and an unlimited set of coding modes.

5.4 RDC result validation in HM 11.0

Further verification of the RDC results was obtained by running the proposed schemes in the most up-to-date version (at the time of development of the work) of the reference encoder, HM 11.0.

Table 5.4 *RDC Comparison of the Proposed QP-Specific (S_{26}-S_{28}) and QP-Independent (S_{14}-S_{17}) Schemes in HM 8.0 and HM 11.0.*

Cfg.	ID	N_{SMP} / N_{AMP} QP 22	QP 27	QP 32	QP 37	**BD-rate HM 8.0 / 11.0**	$\Delta\ Mcpf_{enc}$ **HM 8.0**	**HM 11.0**
RA	S_{14}	$\{4^d,8^d,16^d,32^d\}$ / $\{8,16,32\}$				0.2%	-31%	-34%
	S_{15}	$\{4^d,8^d,16^d\}$ / $\{8,16,32\}$				0.3%	-35%	-37%
	S_{16}	$\{4^d,8^d\}$ / $\{8,16,32\}$				0.5%	-37%	-40%
	S_{17}	$\{4^d\}$ / $\{8,16,32\}$				0.9%	-41%	-43%
	S_{26}	$\{4^d,8^d\}$ / $\{8\}$		$\{4^d,8^d,16^d\}$ / $\{8,16\}$	$\{8,16,32\}$	0.5%	-42%	-44%
	S_{27}	$\{4^d\}$ / $\{8\}$	$\{4^d,8^d\}$ / $\{8\}$	$\{4^d,8^d,16^d\}$ / $\{8\}$		0.9%	-47%	-48%
	S_{28}	$\{4^d\}$ / $\{8\}$		$\{4^d,8^d\}$ / $\{8\}$		1.3%	-50%	-51%
LB	S_{14}	$\{4^d,8^d,16^d,32^d\}$ / $\{8,16,32\}$				0.3%	-32%	-36%
	S_{15}	$\{4^d,8^d,16^d\}$ / $\{8,16,32\}$				0.4%	-36%	-40%
	S_{16}	$\{4^d,8^d\}$ / $\{8,16,32\}$				0.7%	-40%	-43%
	S_{17}	$\{4^d\}$ / $\{8,16,32\}$				1.1%	-44%	-46%
	S_{26}	$\{4^d,8^d,16^d\}$ / $\{8\}$	$\{4^d,8^d\}$ / $\{8,16,32\}$	$\{4^d,8^d,16^d\}$ / $\{8,16,32\}$	$\{4^d,8^d,16^d,32^d\}$ / $\{8,16,32\}$	0.5%	-39%	-42%
	S_{27}	$\{4^d,8^d\}$ / $\{8\}$	$\{4^d,8^d,16^d\}$ / $\{8\}$	$\{4^d,8^d\}$ / $\{8,16,32\}$		0.9%	-44%	-47%
	S_{28}	$\{4^d\}$ / $\{8\}$	$\{4^d,8^d\}$ / $\{8\}$	$\{4^d\}$ / $\{8,16,32\}$		1.3%	-49%	-50%

[d] Disable SMPs if $M''_h = Skip$ (Fig. 4.7).

Table 5.4 reports the RDC characteristics of QP-independent (S_{14}-S_{17}) and QP-specific schemes (S_{26}-S_{28}) in HM 8.0 and HM 11.0. BD-rate performance of all schemes remains the same between HM 8.0 and HM 11.0, so complexity is the only changed aspect. In terms of complexity, HM 11.0 implementation gain 2 percentage points in the RA case and 3 percentage point in the LB case over HM 8.0. S_{14}-S_{17} reduce the complexity of the default configuration of HM 11.0 by 34-43% and S_{26}-

S_{28} by 44-51% in the RA case. In the LB case, the respective values are 36-46% and 42-50%. Generally speaking, switching from HM 8.0 to HM 11.0 has no notable effect on the relative RDC performances of the proposed and previous schemes.

6. CONCLUSIONS

This Thesis explored the most complex part of HEVC, Inter prediction, whose RDC characteristics were analyzed for the purpose of finding efficient optimizations. In practice, the optimizations were done by exploring parameterization of the Inter prediction mode and block partitioning structure. First, a coarse refinement of the parameters was done and the results were used to select the following optimization steps until the most suitable parameters were found.

The analysis was done with the HEVC reference encoder (HM), versions 8.0 and 11.0. The tests were conducted with HEVC test video sequences of resolutions $\in$ {240p, 480p, 720p, 1080p, 1600p} with QP $\in$ {22, 27, 32, 37} under the RA and LB coding configurations. Objective quality measurement, PSNR, was used for distortion analysis and Intel VTune provided the cycle-level complexity results. Different test cases increased the number of tests up to multiple thousands. This made test setups with a few computer unfeasible and led to the creation of a new software solution to automatically run the needed tests.

TUT Task Manager, a customized software for multi-computer task processing was designed to run tests in large scale without limiting the testing only to the specific case of the parametric exploration. It gained time by automating crucial parts of the analysis and enabled deeper study of the proposed schemes presented in this Thesis. Easy adaptation to other test cases allowed an easy change of the HM version in the later stage of the analysis, and even Intel VTune analysis could be completed using the same solution.

HM MP was used as a base for the optimizations. The coding tree structure and the square motion partitions were adopted directly from HM MP because of their key role in encoding. Intra and Skip modes were also left as-is because of their large RD performance gain and minor impact on complexity. A target was to keep the bit rate overhead of the proposed schemes at around 1% compared to HM MP. This work seeked to optimize SMP and AMP mode structures for finding feasible

optimizations, since their complexity impact in the HM MP encoder was found to be 60%. They are able to give the bit rate gains of 3.8% and 4.5% in the RA and LB cases, respectively. The target was set to minimize the complexity and keeping the bit rate gains as high as possible.

Three techniques were used to deal with the complexity overhead of the SMP and AMP mode evaluation:

1. Conditional evaluation for the SMP modes
2. The range limitations primarily in the SMP sizes and secondarily in the AMP sizes
3. Selection of the ranges of SMP and AMP as a function of the QP

First two items in the list are utilized in the QP-independent schemes S_{14}-S_{17}. These schemes are refined for QP-specific schemes S_{26}-S_{28} using all three techniques in the list. The proposed schemes reduce the complexity of HM MP encoder by 31-51% and 32-50% in the RA and LB case, respectively. The respective bit rate penalties are 0.2-1.3% and 0.3-1.3%. The QP-independent schemes reduce complexity by up to 44% and 46% in the RA and LB case, respectively, after which the additional reduction is obtained through QP-specific schemes. Compared with the prior-art schemes such as ECU+ESD+CFM, the proposed schemes in this work provide better BD-rate performance with the same complexity. These optimizations have been made in a hardware friendly manner using the similar control structures as the original implementations. However, parallelizations and hardware acceleration are not considered when incorporating these schemes in such implementations.

The tools and methods presented in this work can be also utilized in the future studies. Possible targets include the other parts of the video codec, as most of them can be configured with multitude of parameters. Another possibility is to combine the proposed schemes with related works and improve RDC characteristics of HM further.

BIBLIOGRAPHY

[1] G. J. Sullivan, "Overview of international video coding standards," *ITU-T VICA Workshop*, July 2005, [Online]. Available: http://www.itu.int/ITU-T/worksem/vica/docs/presentations/S0_P2_Sullivan.pdf.

[2] *ITU-T Recommendation H.264, "Advanced video coding for generic audiovisual services,"*, international Telecommunication Union, Mar. 2009.

[3] 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.

[4] J. Vanne, M. Viitanen, T. D. Hamalainen, and A. Hallapuro, "Comparative rate-distortion-complexity analysis of HEVC and AVC video codecs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1885–1898, Dec. 2012.

[5] J. Vanne, M. Viitanen, and T. D. Hämäläinen, "Efficient mode decision schemes for HEVC inter prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 9, pp. 1579–1593, Sep. 2014.

[6] H. Kalva, "The H.264 video coding standard," *IEEE MultiMedia*, vol. 13, no. 4, pp. 86–90, Oct. 2006.

[7] J. R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards - including high efficiency video coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1669–1684, Dec. 2012.

[8] I. K. Kim, J. Min, T. Lee, W. J. Han, and J. Park, "Block partitioning structure in the HEVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1697–1706, Dec. 2012.

[9] Y. Yuan, I. K. Kim, X. Zheng, L. Liu, X. Cao, S. Lee, M. S. Cheon, T. Lee, Y. He, and J. H. Park, "Quadtree based nonsquare block structure for inter frame coding in high efficiency video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1707–1719, Dec. 2012.

[10] P. Helle, S. Oudin, B. Bross, D. Marpe, M. O. Bici, K. Ugur, J. Jung, G. Clare, and T. Wiegand, "Block merging for quadtree-based partitioning in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1720–1731, Dec. 2012.

[11] C. M. Fu, E. Alshina, A. Alshin, Y. W. Huang, C. Y. Chen, C. Y. Tsai, C. W. Hsu, S. M. Lei, J. H. Park, and W. J. Han, "Sample adaptive offset in the HEVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1755–1764, Dec. 2012.

[12] *HM (HEVC Test Model)*, [Online]. Available: https://hevc.hhi.fraunhofer.de/.

[13] *HOMER (Hevc Open Mpeg EncodeR)*, [Online]. Available: http://homerhevc.com/.

[14] M. Viitanen, A. Koivula, A. Lemmetti, A. Ylä-Outinen, J. Vanne, and T. D. Hämäläinen, "Kvazaar: open-source HEVC/H.265 encoder," in *Proc. ACM Int. Conf. Multimedia*, Amsterdam, The Netherlands, Oct. 2016. [Online]. Available: http://doi.acm.org/10.1145/2964284.2973796

[15] A. Koivula, M. Viitanen, J. Vanne, T. D. Hämäläinen, and L. Fasnacht, "Parallelization of Kvazaar HEVC intra encoder for multi-core processors," in *Proc. IEEE Workshop on Signal Process. Systems*, Hangzhou, China, Oct. 2015.

[16] M. Viitanen, A. Koivula, J. Vanne, and T. D. Hämäläinen, "Kvazaar HEVC still image coding on Raspberry Pi 2 for low-cost remote surveillance," in *Proc. Visual Communications and Image Process.*, Singapore, Dec. 2015.

[17] A. Koivula, M. Viitanen, A. Lemmetti, J. Vanne, and T. D. Hämäläinen, "Performance evaluation of Kvazaar HEVC intra encoder on Xeon Phi many-core processor," in *Proc. IEEE Global Conference on Signal and Information Process.*, Orlando, Florida, USA, Dec. 2015.

[18] A. Lemmetti, A. Koivula, M. Viitanen, J. Vanne, and T. D. Hämäläinen, "AVX2-optimized Kvazaar HEVC intra encoder," in *Proc. IEEE Int. Conference on Image Process.*, Phoenix, Arizona, USA, Sep. 2016.

[19] *Turing codec*, [Online]. Available: http://turingcodec.org/.

[20] *x265 HEVC encoder*, [Online]. Available: https://bitbucket.org/multicoreware/x265/wiki/Home.

[21] C. Rosewarne, B. Bross, M. Naccari, K. Sharman, and G. Sullivan, "High efficiency video coding (HEVC) Test Model 16 (HM 16) improved encoder description update 4," *document JCTVC-V1002, Geneva, Switzerland, Oct. 2015.*

[22] K. Choi, S. H. Park, and E. S. Jang, "Coding tree pruning based CU early termination," *document JCTVC-F092, Torino, Italy, Jul. 2011.*

[23] J. Yang, J. Kim, K. Won, H. Lee, and B. Jeon, "Early SKIP detection for HEVC," *document JCTVC-G543, Geneva, Switzerland, Nov. 2011.*

[24] R. H. Gweon and Y.-L. Lee, "Early termination of CU encoding to reduce HEVC complexity," *document JCTVC-F045, Torino, Italy, Jul. 2011.*

[25] F. Bossen, "Common test conditions and software reference configurations," *document JCTVC-J1100, Stockholm, Sweden, Jul. 2012.*

[26] G. Bjøntegaard, "Calculation of average PSNR differences between RD curves," *document VCEG-M33, Austin, TX, USA, Apr. 2001.*

[27] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC complexity and implementation analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, Dec. 2012.

[28] C. E. Rhee, K. Lee, T. S. Kim, and H. J. Lee, "A survey of fast mode decision algorithms for inter-prediction and their applications to high efficiency video coding," *IEEE Trans. Consumer Electron.*, vol. 58, no. 4, pp. 1375–1383, Nov. 2012.

[29] G. Correa, P. Assuncao, L. Agostini, and L. A. da Silva Cruz, "Performance and computational complexity assessment of high-efficiency video encoders," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1899–1909, Dec. 2012.

[30] X. Shen, L. Yu, and J. Chen, "Fast coding unit size selection for HEVC based on Bayesian decision rule," in *Proc. Picture Coding Symp.*, May 2012.

[31] L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An effective CU size decision method for HEVC encoders," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 465–470, Feb. 2013.

[32] H. L. Tan, F. Liu, Y. H. Tan, and C. Yeo, "On fast coding tree block and mode decision for high-efficiency video coding (HEVC)," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Process.*, Mar. 2012.

APPENDIX A. TUT TASK MANAGER PROTOCOL EXAMPLE

```
/*
0x04 request file (client -> server)
  ID (uint32_t)
  name (String)
  resume (uint64_t)
  */
QByteArray clientProtocol::requestFilePacket(unsigned int ID,
   ↪ QString name, unsigned long long resume)
{
    QByteArray data;
    int curpos = 0;
    data.push_back((char)0x04);
    curpos += 1;
    data.push_back((char)0); data.push_back((char)0); data.
       ↪ push_back((char)0); data.push_back((char)0);
    qToBigEndian(ID,(uchar *)data.data()+curpos);
    curpos += 4;
    data.push_back((char)0); data.push_back((char)0);
    qToBigEndian((quint16)name.size(),(uchar *)data.data()+
       ↪ curpos);
    curpos += 2;
    for(int i = 0; i < name.size(); i++)
    {
        data.push_back(name[i].toLatin1());
    }
    curpos += name.size();

    data.push_back((char)0); data.push_back((char)0);
    data.push_back((char)0); data.push_back((char)0);
    data.push_back((char)0); data.push_back((char)0);
    data.push_back((char)0); data.push_back((char)0);
    qToBigEndian((unsigned long long)resume,(uchar *)data.data()
       ↪ +curpos);
    curpos += 8;
```

```
    qDebug () << "requestFilePacket";
    return data;
}

/*
0x04 request file (client -> server)
  ID (uint32_t)
  name (String)
  resume (uint64_t)
*/
int serverProtocol::readRequestFilePacket(QByteArray &data,int &
   ↪ len,int &ID, QString &name, unsigned long long &resume)
{

    int curpos = 1;
    if(data.size() < 14)
    {
        return NEED_MORE_DATA;
    }
    ID = qFromBigEndian<quint32>((const uchar *)data.data()+
       ↪ curpos);
    curpos += 4;

    name.clear();
    int namelen = qFromBigEndian<quint16>((const uchar *)data.
       ↪ data()+curpos);
    curpos += 2;

    if(data.size() < curpos+namelen+8)
    {
        return NEED_MORE_DATA;
    }

    for(int i = 0; i < namelen; i++)
    {
        name+=QChar((char)data[curpos+i]);
    }
    curpos += namelen;
```

```
    resume = qFromBigEndian<quint64>((const uchar *)data.data()+
        ↪ curpos);
    curpos += 8;

    len = curpos;

    return PACKET_OK;
}
```