A Framework for Video Coding Analyzer

Sakshi Jain

ECE

IIIT-Delhi

Under the Supervision of Dr.Alexander Fell

A thesis submitted for the degree of

Master of Technology

2016 June

1. Reviewer: Name

2. Reviewer:

Day of the defense:

Signature from head committee:

Abstract

A video analyzer is a comprehensive bitstream analysis tool which accelerates development and debugging of video bitstreams while ensuring compliance with industry standards. There are many conventional analyzers present for different video standards like H.264, HEVC which are compliant only with the respective video sequence format. In this work, a generalized analyzer is proposed which is flexible enough to incorporate any video codec, as it is video syntax independent. It also integrates encoder which helps in the development and analysis of video encoder. In a case study, the analyzer is tested for VP9 bitstreams which is the latest video coding format.

Acknowledgements

I express my sincere gratitude to my thesis supervisor Dr. Alexander Fell and Mr. Ajit Singh Motra for their valuable guidance and inspiration throughout my thesis work. I appreciate the way they helped me at each step of my work and the unwavering faith they showed in my abilities.

I would like to thank Mr. Mahesh Narain Shukla for his guidance and help in all matters without which this work couldnt have been turned into reality.

I would like to thank all my friends, and would remain indebted to them for the way they rejoiced in my happiness and backed me in difficult times. I would also like to thank every person who has directly or indirectly helped me in the completion of my work.

Last, but not the least, I would like to express my gratitude towards my family which has always supported and encouraged me in every endeavor of mine.

Sakshi Jain

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Due to increasing usage of videos online on websites like YouTube, the demand for the high quality video content and reduced bitrate increases as most of the traffic on the Internet is video today. Thus compression is an essential technology in the video industry. Hence, new compression standards are evolving aiming to reduce bitrate by more than 50% preserving the same video quality as of the previous standards or enhancing video quality at the same bitrate. Video compression is comprised of a series of steps as detailed in [1], maintaining an appropriate balance between coding efficiency and quality of the video. Several state-of-the-art compression standards evolved starting from H.261, MPEG-1, MPEG-2/H.262, H.263 and MPEG-4 to H.264/MPEG-AVC [2], H.265/MPEG-HEVC [3] and VP9 [4]. VP9 is the latest video coding format developed by Google as a part of the WebM project [5] and its bitstream syntax was finalized and released in June 2013. It is royalty free (unlicensed) and has improved coding tools and bitstream features as explained in [6].

While encoding files or during their compression, errors can occur which result in an erroneous bitstream. To detect these errors the user has to read all necessary coding parameters and coefficients from a log file which is a very time consuming and tedious task. The bitsream analyzer is a tool through which the user can analyze the bitstream in an effective Graphical User Interface (GUI) environment and in lesser time duration. Various analyzers have been developed in the past. HEVC analyzers include Elecard HEVC analyser [7], Parabola Explorer [8] and Zond265 [9]. Vega [10], CodecVisa [11] and Intel VPA [12] support H.264, HEVC and VP9. These conventional analyzers have dedicated decoders inside them which can support a particular bitstream format [13].

The compressed sequence is sent to the decoder and an error is generated, if the bitstream syntax is not compatible with the decoder. If the syntax is decoder compatible, it decodes the bitstream and generates parameters like mode, transform unit (TU) size, prediction unit (PU) size, motion vectors (MV), etc. which are visualized through a GUI of the analyzer.

As new video standards are constantly evolving, there is a need for analyzers to incorporate these changes. [13] introduces an analyzer which is being tested for H.265 bitstreams and could be extended for other video standards. In [14], an HEVC analyzer has been developed for rapid prototyping including encoder side behavior. In this thesis, a novel video analyzer is presented which is decoder independent and only depends on the parameter files for data extraction. As these files can be generated from any video codec, the analyzer is easily adaptable to all video standards. Unlike previous analyzers, the proposed analyzer also integrates an encoder to check the performance of the codec which helps in the development and improvement of codec parameters. The simple and structured implementation of this analyzer together with the integrated encoder makes it a favorable, flexible and powerful tool.

Presently, considerable research is being carried out for the development of VP9 analyzers due to the world-wide acceptance of the VP9 codec for its efficiency and royalty-free quality. Thus, the analyzer is developed and tested for VP9 bitstreams and its bitstream features are displayed on the screen of the analyzer in a user friendly way. The software implementation of VP9 codec is taken from git repository [15]. This paper is divided into 4 sections. Section 2 gives the overview of the analyzer and the main features it supports. The screenshots of the VP9 analyzer and the visual display are explained in section 3. Section 4 compares the proposed analyzer with the other analyzers. Section 5 concludes the paper and discusses future work.

Chapter 2

Overview of the Analyzer

The complete overview of the analyzer and its features have been detailed in Fig. 2.1. The compressed bitstream to be analyzed is sent to the decoder. The decoder generates a log file containing all necessary coding parameters and a corresponding uncompressed yuv file. The analyzer utilizes these files to graphically display all the features of a compressed bitstream by navigating through frames of the video sequence. The analyzer is capable of overlaying intra and skip blocks over the selected frame and producing the partition statistics exhibiting the number of different types of inter (P) and intra (I) block partitions.

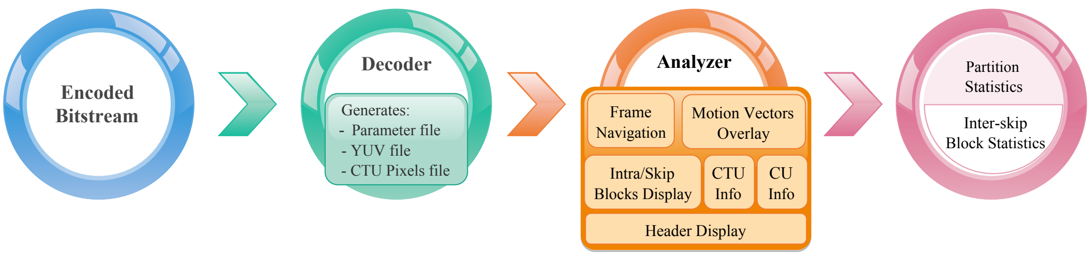


Figure 2.1: Overview of the Analyzer

The main features of the proposed analyzer are:

- Generalized architecture
- Quick, simple and easy access of all coding parameters
- Supports 4:2:2, 4:4:4 along with 4:2:0 chroma subsampling

- Supports all bit-depths: 8, 10 and 12
- Integrated Encoder

These contributions are described in detail in the following sections.

2.1 Generalized Architecture

To generalize the analyzer to ensure support for future video codecs, the following tasks are implemented:

2.1.1 Grouping of Parameters

The parameters are grouped into two categories, namely Frame Level and CU (Coding Unit) Level parameters such that any video codec can be incorporated by categorizing its parameters into the levels shown in Table 2.1. The probabilities of all coding parameters are included in the frame header which are used to decode the current frame. Y, U and V modes indicate the prediction modes subject to the block mode (Inter or Intra). The number and types of Inter and Intra modes differ depending on the video standards. For example: HEVC has 35 modes of intra prediction whereas VP9 has only 10 modes.

2.1.2 Extraction of Parameters

The HEVC has CUs of pixel sizes 16×16, 32×32 and 64×64 i.e. square sized blocks only. Whereas in VP9, the CU can be of any size from 64×64 down to 8×8 including rectangular block sizes. Thus, every video standard can have different types of partitions. As the minimum CU size can be 8×8 pixels, for the analyzer to support any video codec, every block is treated as multiple 8×8 CUs for the ease of data parsing. For example, a 32×16 CU block is saved as eight 8×8 similar sub-blocks as shown in Fig. 2.2 with the x and y coordinates of the top left corner of the block with respect to the frame so that the data can be extracted from the parameter file on the basis of its location. If (x,y) is the location of a parent block of size $w \times h$, then its sub-block locations are given as $(x+8m,y+8n)$ where $0 \leq n \leq (h/8-1)$ and $0 \leq m \leq (w/8-1)$. The CU information of the parent block is replicated for all sub-blocks which is stored along

Table 2.1: Coding Parameters

Frame Level	• Resolution: width × height of the frames • Profile: 4:2:0, 4:2:2 or 4:4:4 with bit-depths 8/10/12 • Probabilities of all the parameters • Filter, quantization and segmentation parameters
CU Level	• Block mode: Inter or Intra • TU size: 4×4, 8×8, 16×16 or 32×32 • PU size: 64×64 downto 4×4 • Y/U/V modes: Inter or Intra modes • MV: Displacement in x and y directions • Skip: CU skipped or not

with its location in the parameter file. Table 2.2 shows the number of similar 8×8 blocks to be saved for different block partitions.

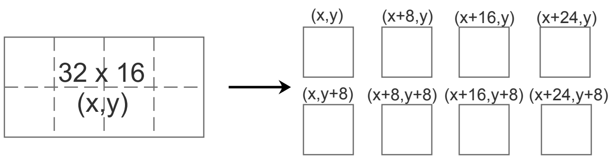


Figure 2.2: 32×16 Block Information saved as eight 8×8 Blocks

2.1.3 Integration of the Encoder

Each video standard provides various coding tools to encode a video file. There is a provision to pass configuration parameters to the analyzer to encode the yuv file and the resultant compressed bitstream is saved in the current directory. This feature assists in obtaining the compressed bitstream, if the original source video file is available to the user and the encoder performance can be checked visually by extracting the parameters of the compressed file. Fig. 2.3 shows a configuration example for VP9 encoding which has to be passed to the analyzer to get the compressed bitstream. The detailed explanation of all VP9 encoding parameters is given in [17].

Table 2.2: Extraction and Saving of Blocks as 8×8

Partitions	Number of 8×8 blocks
8×8	1
8×16, 16×8	2
16×16	4
16×32, 32×16	8
32×32	16
32×64, 64×32	32
64×64	64

```
--good --cpu-used=0 threads=0 profile=0
--aq-mode=0
--fps= <FrameRate>
--end-usage=3 --cq-level=<QP>
--kf-max-dist= <IntraPeriod> --kf-min-dist= <IntraPeriod>
--minsection-pct=0 --maxsection-pct=2000
--auto-alt-ref= 1
--arnr-maxframes=7 --arnr-strength=5
--codec=vp9 -v -t 0 -w<Width> -h<Height> -p 2
```

Figure 2.3: Configuration Parameters for VP9 Encoder

2.2 Organization of Files

When yuv file is analyzed, the following files are saved in the current project directory in one folder which are used as data to be displayed on the analyzer GUI. Each file has its own significance for presenting visual information to the user.

1. Output parameter file: It consists of all essential parameters and coefficients evaluated in the decoding process which are required to analyze the video bitstream in detail. The file has multiple sections equal to the number of frames in the video sequence. Each frame section consists of two parts: frame header (Frame-Level information) and compressed frame data (CU-Level information). The required parameter is extracted using data parser by scanning through this text file, and

is displayed in the analyzer GUI.

2. YUV file: This file is generated after decoding and is used to construct and display frames.

3. CTU (Coding Tree Unit) pixel files: The pixel values of each frame are rearranged in the form of 64×64 pixel block sizes, i.e. a CTU and are stored in separate files. These values are displayed when the mouse pointer hovers over the frame.

Chapter 3

Case Study - VP9 Analyzer

A snapshot of the complete VP9 analyzer is shown in Fig. 3.1. The analyzer is well structured as all required information is divided in multiple resizable windows. The figure illustrates that multiple video sequences can be explored simultaneously. The extraction and display of all information is very quick and user does not have to wait even for large frame sizes. Table 3.1 describes all windows and the important features of the GUI.

Table 3.1: Features of GUI

Window 1	Frame navigation for random access of frames
Window 2	Frame number and image of selected frame
Window 3	Video sequence summary; resolution, format and bit-depth
Window 4	Parameters of compressed and uncompressed headers
Window 5	CTU information, image and CU information
Window 6	Partition and skip blocks statistics
Intra/skip Blocks	Blocks to be displayed in different colors
MV Overlay	To display the movement of the object
Mouse Hover	Display of YUV pixels instantaneously

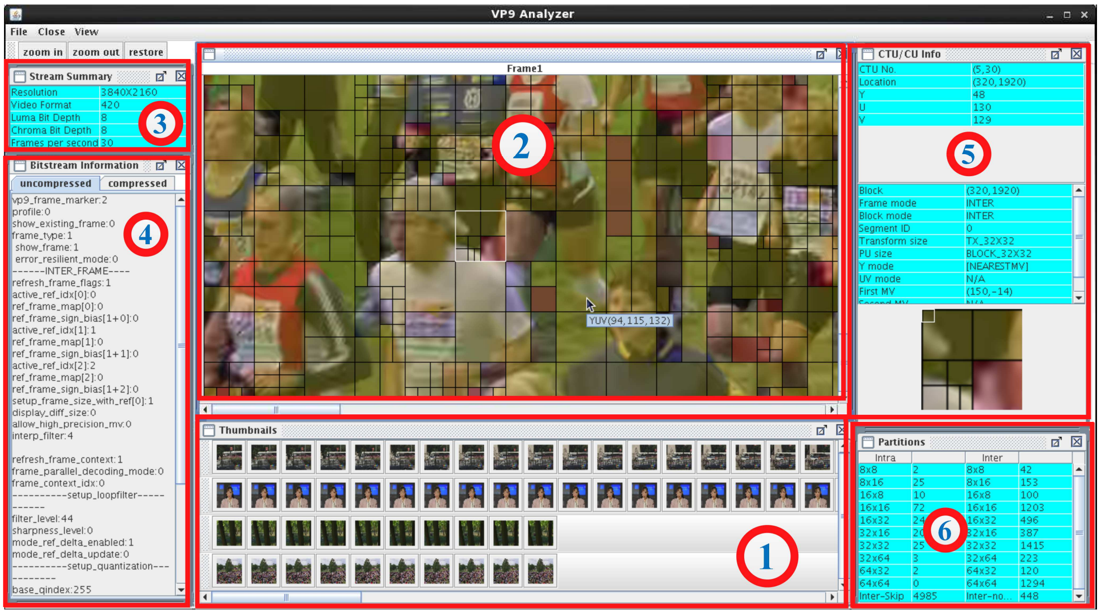


Figure 3.1: Snapshot of the Analyzer

The user can explore a particular frame by navigating through the sequence of frames in a video stream. When a frame is selected, its image and number in the sequence are displayed in window 2 and the frame can be analyzed on a CTU level. Window 4 shows the parameters of the compressed and uncompressed headers, i.e. the complete syntax information which is elaborated in [16]. When one CTU is selected inside the frame, its information is displayed in the first table of window 5 and image at the bottom as shown in Fig. 3.1. The CTU number indicates its location with respect to the number of CTUs, Location indicates its coordinates with respect to the frame and YUV pixels are the pixels of the top-left corner of the selected CTU.

The CTU can be further explored on a CU level whose parameters are listed in second table of window 5. The size of PU is equal to CU for size greater than 8×8 and 4×4, 4×8 or 8×4 for 8×8 CU. Thus, more than one Y mode is displayed when the PU size is less than 8×8, i.e. two Y modes for 8×4 and 4×8 and four Y modes for 4×4. There are four inter modes of prediction which VP9 supports: NEARESTMV, NEARMV, ZEROMV and NEWMV along with 10 modes for intra prediction, namely TM pred (True-Motion prediction), DC pred (DC prediction) and 8 directional modes (horizontal, vertical, D27, D153, D135, D117, D63 and D45). The first MV is always displayed for inter frame mode and the second MV is shown only in case of compound prediction.

3.0.1 Test sequences

The analyzer has been tested with several test sequences downloaded from [18] covering all scenarios. Video sequences of different resolutions and bit-depth (8/10) have been analyzed including HD (1920×1080) and 4K UHD (3840×2160). Table 3.2 shows the VP9/WebM bitstreams that have been tested using this analyzer.

Table 3.2: Test Sequences

Name	Resolution
Park_joy	160×90
Bus	176×144
Stefan	202×198
Blue_sky	352×288
Akiyo	352×288
Pedestrian_area	1920×1080
Crowd_run	3840×2160

3.0.2 Overlay Feature

To make the analyzer more elaborative, four types of overlays are added in the view menu: Grid, Motion Vectors, Intra Blocks and Skip Blocks which can be overlayed individually or in combination of any number of views. Fig. 3.2 shows the combination of grids, intra and skip blocks with intra blocks in red color and skip blocks in yellow. Fig. 3.3 shows the overlayed motion vectors which represents the motion of objects in a particular direction.

3.0.3 Statistics Display

The number of different types of block partitions for Inter and Intra modes are calculated and displayed in the partitions window as shown in Fig. 3.4. The total number of

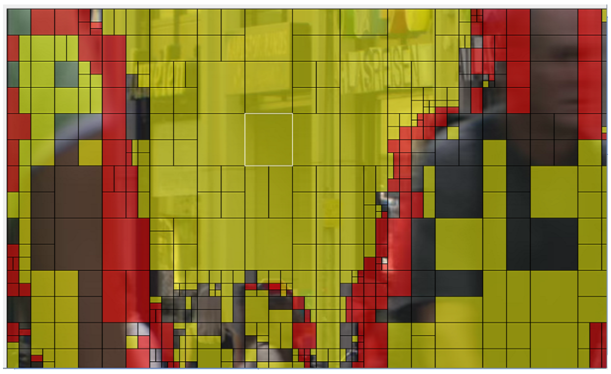


Figure 3.2: Intra and Skip Blocks Overlay

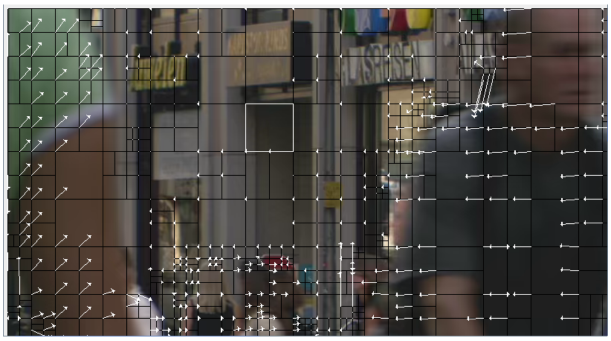
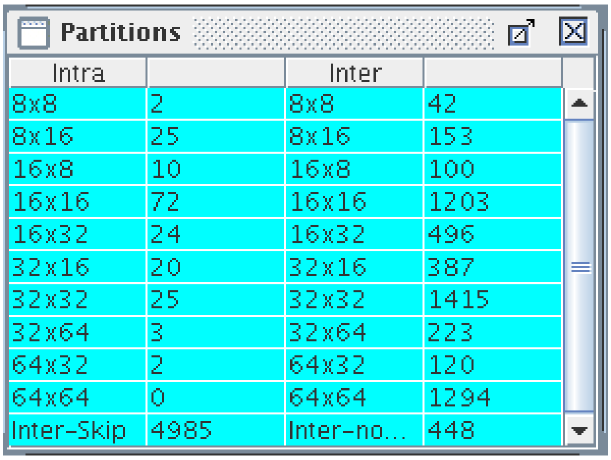


Figure 3.3: Motions Vectors Overlay

Inter skip and non-skip blocks are also displayed to help the user to improve the efficiency of the compressed sequence. These distributions help in redesigning a sequence with a shorter codeword for frequently occurring block partitions of a particular mode to increase the efficiency and to lower the bitrate.



Partitions

Intra		Inter	
8x8	2	8x8	42
8x16	25	8x16	153
16x8	10	16x8	100
16x16	72	16x16	1203
16x32	24	16x32	496
32x16	20	32x16	387
32x32	25	32x32	1415
32x64	3	32x64	223
64x32	2	64x32	120
64x64	0	64x64	1294
Inter-Skip	4985	Inter-no...	448

Figure 3.4: Partitions and Skip Blocks Statistics

Chapter 4

Comparison

The main commercially available analyzers which support VP9 standard along with other standards are Intel VPA [12], Vega [10] and CodecVisa [11]. Table 4.1 compares the features supported by the developed analyzer with these analyzers. There are some features available in the other analyzers which the proposed analyzer does not support. For instance, the detailed view of loop filter strength of each block in a frame, heat map which shows the distribution of compressed bits among different blocks of a frame and efficiency map which shows the arithmetic coding efficiency of each block visually. All these additional views can be easily incorporated in the proposed analyzer as all the required information is saved in the log files. Unlike other analyzers, the proposed analyzer with integrated encoder is generic which can support next generation video codecs.

Table 4.1: Comparison with other Analyzers

Features	Proposed Analyzer	Intel VPA	CodecVisa	Vega
Codecs supported	Generic - support any codec	MPEG-2, H.264, HEVC and VP9	MPEG-2, H.264, HEVC and VP9	MPEG-2, H.264, HEVC and VP9
Integrated Encoder	Yes	No	No	No
Profiles	All Profiles	All Profiles	All Profiles	All Profiles
Overlays	Partitions, I/P, skip blocks, MV	Partitions, I/P, skip blocks, MV	Partitions, I/P, skip blocks, MV	Partitions, I/P, skip blocks
Header Information	Organized as compressed and uncompressed header	Organized according to the information type	As a frame header tree	Display both headers
Statistics (Number of I/P blocks)	Yes, along with skip blocks	Yes	No	No
Other Views	-	Heat, PSNR and efficiency views	Bit numbers for each block	Loop filter process view

Chapter 5

Conclusion

In this work, a video coding analyzer is developed that extracts in-depth information of all video frames as frame-level and block-level information from the decoder which is used as the data to be displayed on the visual screen to have detailed insight into the codec workings. The developed analyser is independent of bitstream syntax, hence it can be generalized to support any bitstream. The analyzer supports all bit-depths with all types of chroma sub-sampling formats. With the integrated encoder support and the ability to display all coding parameters and statistics graphically over the video content makes this analyzer an appropriate choice for the codec designers. This novel analyzer can be used for quick and easy inspection of encoder performance and can also help engineers in the testing and development phases of video codecs.

Bibliography

[1] How Video Compression Works, http://www.bdti.com/MyBDTI/pubs/video_compression.pdf. 1

[2] Thomas Wiegand, Gary J. Sullivan, Gisle Bjontegaard, Ajay Luthra. *"Overview of the H.264/AVC Video Coding Standard,"* IEEE Trans. on Circuits and Systems for Video Technology,vol. 13 No. 7, Jan 2011. 1

[3] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand, *"Overview of the High Efficiency Video Coding (HEVC) Standard,"* IEEE Trans. on Circuits and Systems for Video Tech., vol. 22, No. 12, Dec 2012. 1

[4] *How VP9 works, technical details & diagrams,* http://forum.doom9.org/showthread.php?t=168947. 1

[5] The WebM Project, *WebM: an open web media project*, http://www.webmproject.org/, 2014. 1

[6] D. Mukherjee, J. Bankoski, A Grange, H. Jingning, J. Koleszar, P. Wilkins, X. Yaowu, and R. Bultje, *"The latest open-source video codec VP9 - an overview and preliminary results,"* in Proc. PCS, San Jose, USA, Dec 2013, pp. 390393. 1

[7] Elecard, *Elecard HEVC bitstream analyzer*, www.elecard.com. 1

[8] Parabola Research, *Parabola HEVC bitstream analysis software,* www.parabolaresearch.com. 1

[9] Solveigmm, *Zond 265: HEVC bitstream analyzer*, www.solveigmm.com/en/products/zond. 1

[10] Interra Systems, *Vega: H.264,HEVC and VP9 analyzer*, www.interrasystems.com/analyzers.php.

1, 15

[11] Codecian, *CodecVisa: HEVC and VP9 bitstream analyzer*, www.codecian.com.

1, 15

[12] Intel, *Intel Video Pro Analyzer*, https://software.intel.com/en-us/intel-video-pro-analyzer. 1, 15

[13] Taoran Lu, Xiaoan Lu, Qian Xu, Yunfei Zheng, Joel Sole and Peng Yin, *"A Video Coding Analyzer for Next-Generation Compression Standards,"* IEEE International Conference on Consumer Electronics (ICCE), 2011. 1, 2

[14] D. Springer, W. Schnurrer, A. Weinlich, A. Heindel, J. Seiler and A. Kaup, *"Open source HEVC analyzer for rapid prototyping (HARP),"* in Proc. IEEE International Conference on Image Processing (ICIP), Paris, France, Oct. 2014. 2

[15] https://github.com/webmproject/libvpx. 2

[16] Adrian Grange, Peter de Rivaz and Jonathan Hunt, *VP9 Bitstream & Decoding Process Specification*, https://storage.googleapis.com/downloads.webmproject.org/docs/vp9/vp9-bitstream-specification-v0.6-20160331-draft.pdf

10

[17] Encode Parameter Guide, http://www.webmproject.org/docs/encoder-parameters/. 5

[18] "Derfs dataset," https://media.xiph.org/video/derf/. 11