2024/07/09 Research Report

RTCL Lab Meeting
Presenter: Chaehoon Park

Intro

Background

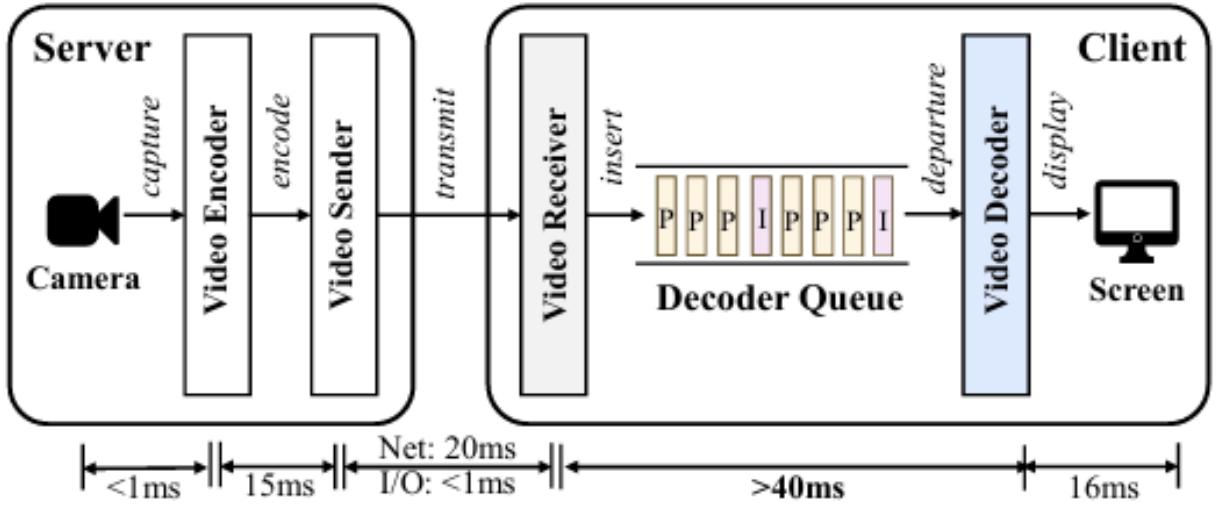
AR/VR: a powerful tool across various sectors, enhancing education, training, healthcare, entertainment

Real-time high-quality (HQ) videos are more and more popular in daily life (e.g., 4K video conferencing, AR/VR)

“Argus: Real-Time HQ Video Decoding with CPU Coordinating on Consumer Devices”

About Paper

Problem Definition



The encoding delay is light since videos are processed on commercial servers.

With the development of communication techniques and I/O protocols (e.g., 5G/6G, WiFi-6), the frames can be delivered within tens of microseconds

→ Bottleneck is located at the decoding phase.

About Paper

Thinkable Solutions

Sol 1) using dedicated hardware to decode videos

Vision Pro: Apple developed a customized processor, R1, for the headset.

→ higher energy consumption and economic costs, unsuitable for most consumer devices. processor overheating or network fluctuations can occur.

Sol 2) Reduce latency through dynamic frame skipping

alleviates the pressure on decoder queue latency, but leads to a decline in frame rates.

About Paper

Solution in the paper

CPU resources are not fully used when the VPU is busy

A typical decoding process can be divided into three parts:
a. parsing the bitstream
b. entropy decoding
c. macroblock reconstruction

Argus: new real-time HQ video decoding solution
migrates some frames to the CPU for decoding when the decoder queue is congested.

→ is possible to realize both low latency and high quality, by making use of the wasted CPU resources

About Paper

Challenges of decoder design

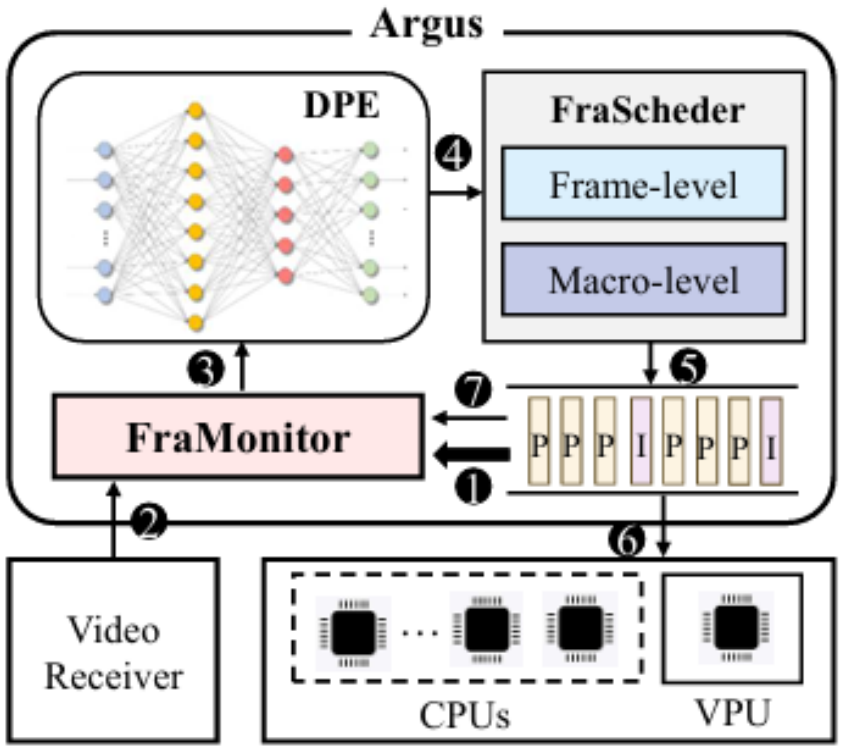
1. It is challenging to estimate the decoder pressure accurately and immediately.
→ decoding speeds of frames with different types and sizes differ

2. Scheduling improper frames to the CPU can degrade, instead of improving the performance
→ because video frames have complicated internal dependencies

3. The scheduling system should be lightweight with negligible overhead

About Paper

Architecture of Argus



DPE (Intelligent decoder pressure estimation model)

: Identify the busy state as early as possible

FraScheder (Frame-characteristics aware scheduler)

: Offloads several frames to CPUs to alleviate the pressure

About Paper

- DPE (Intelligent decoder pressure estimation model)

Takes multidimensional frame information (e.g., frame types, frame size) into account

The decoder's pressure can be estimated by identifying whether a new coming frame's latency becomes high → With NN model

Advantages compared to traditional approach)

Faster than existing control mechanisms that estimate the VPU state based on queue length or queuing delay.

Having more frames in the queue does not necessarily mean higher latency, this problem is effectively addressed by Argus.

About Paper

- DPE (Intelligent decoder pressure estimation model)

1) Input Features and Labels

For every frame, collect five raw fields: the frame type and size, the number of macroblocks and their types, and the number of frames in the queue.

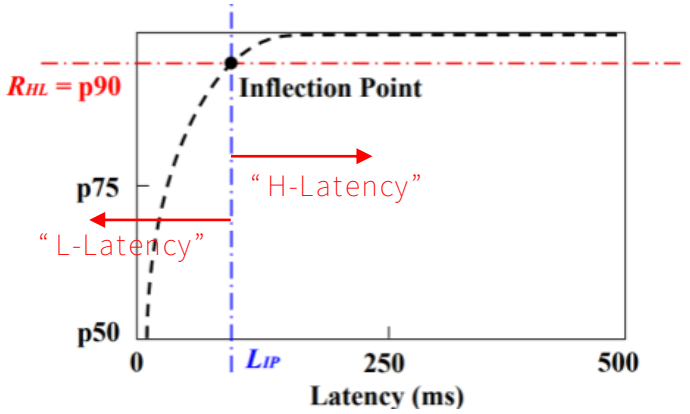
For a new arrival frame, Argus just needs to identify whether it can be quickly processed. If yes, insert it into the decoder queue; if not, wake up the scheduling engine.

→ not labeling with actual end-to-end latency, but makes a binary inference

Label with “ H-Latency” (meaning the decoder is busy) or “ L-Latency” (meaning idle)

About Paper

- DPE (Intelligent decoder pressure estimation model)



2) Offline Training

Train the neural network with labels offline.

The latency distribution of frames reflects the typical latency distributions (Pareto distribution).

Argus separates the low latency ranges(>90%, meet users’ high demand) and high latency ranges(the rest) in a simple approach.

3) Online Inference

After training, the model is deployed in the system for online inference.

Fully connected neural network with three layers, including one input/preprocess layer, one hidden layer, and one output layer.

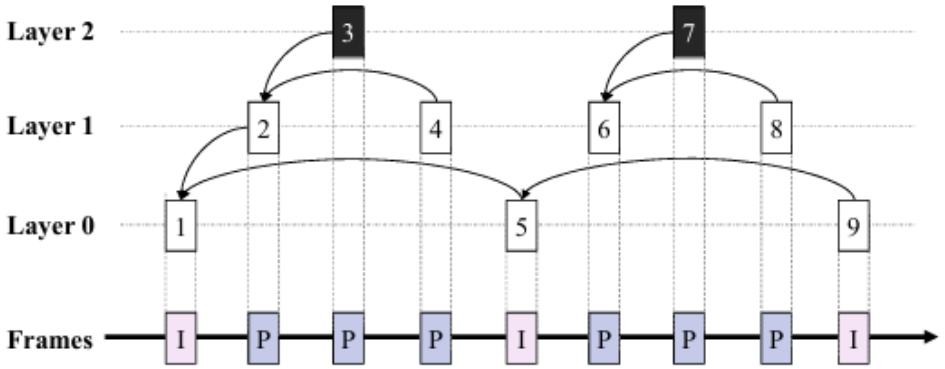
About Paper

- FraScheder (Frame-characteristics aware scheduler)

1) Frame-level Scheduling

Argus performs frame-level scheduling in two steps:

Step 1) Selects the frames that are allowed to be processed in parallelism.



Based on the video protocol, frames on the SVC top layer have no dependency on others, like F3 and F7

→ Argus selects the top layer frames as candidates.

Step 2) Determine whether to offload the candidate.

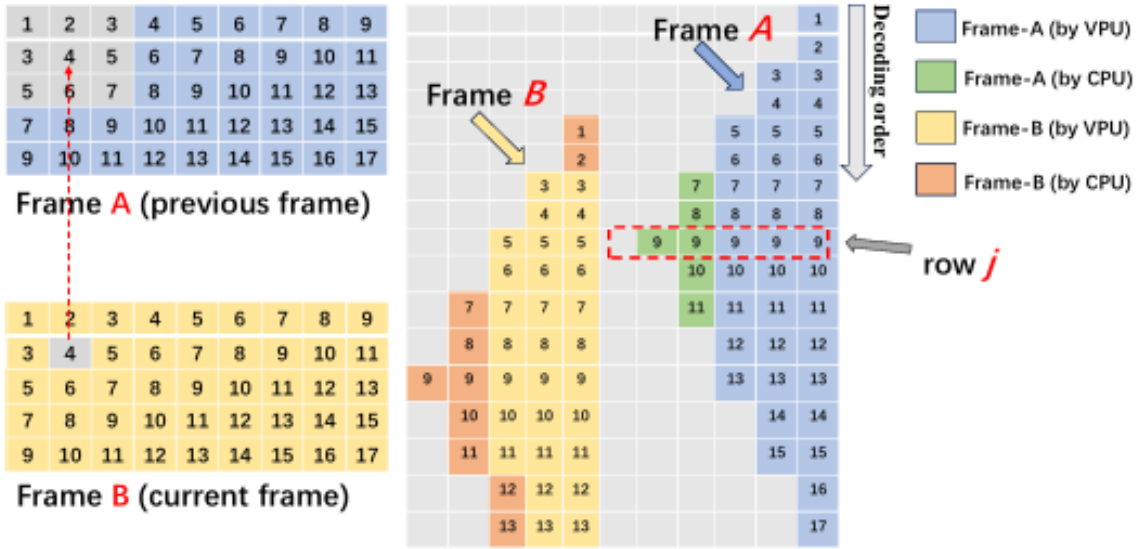
The candidate frame should not be scheduled if it takes too long time to decode on the CPU.

About Paper

FraScheder (Frame-characteristics aware scheduler)

2) Macroblock-level Scheduling

- intra-frame macroblock dependencies
- inter-frame macroblock dependencies



(a) Select the candidates

Classify macroblocks into sets that are independent → schedule them parallely

(b) Determine the scheduling timing

num of macroblocks in row j　　num Cpu + Vpu cores

$$T_j^{cv} = (\lceil R_j^{mb} / (m+n) \rceil) \times T_{mb}$$

the time cost to decode one macroblock

What I am thinking about…

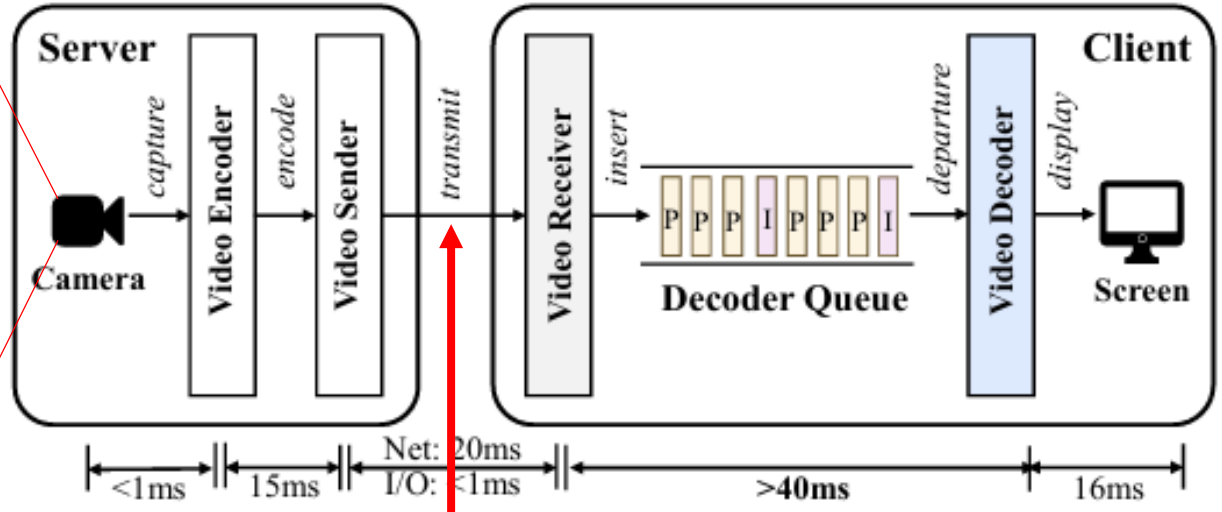
Virtual Conference

VR Games

Applications Etc.

Most of VR applications utilize 3D graphic rendering engine.

What I am thinking about…



Virtual World

3D Graphics Engine
(ex. Unity, Unreal Engine)

Offer Additional Information about frames (or macroblocks) that can help

(1) Identifying decoder pressure
(2) Judging which frames (or macroblocks) to schedule in parallel

What I am thinking about...

- Additional Information that 3D graphics engine can offer

1) Decoding Delay Estimation

Add DPE neural network input features
Serve as scheduling timing in FraScheder

(a) Frames with more complicated contents require more macroblock subdivisions in these details, leading to increased decoding computational complexity.

- Number of distinct objects rendered on the screen
- Frame Complexity Value: ex) Σ((Object Texture Complexity) x (Rendered Area Size))
- Object Texture Complexity: pre-calculated number of macroblocks consisting texture

What I am thinking about…

- Additional Information that 3D graphics engine can offer

1) Decoding Delay Estimation

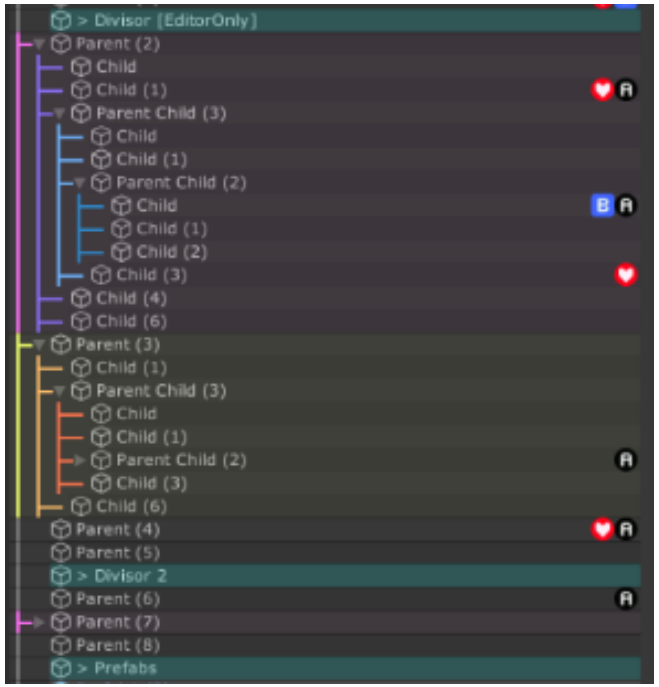
(b) Frames with significant motion or changes require more motion vectors, thus increasing decoding time.

- Camera velocity & angular velocity (+ acceleration, etc.)
- Peripheral Object Transition Value: ex) Σ((Object Velocity) x (Rendered Area Size))

What I am thinking about…

- Additional Information that 3D graphics engine can offer

2) Judging which frames (or macroblocks) to schedule in parallel



3D graphics engine can serve information about dependencies between macroblocks with object hierarchy

Or, reorganize macroblock subdivision in more object-friendly ways
→ reduce number of macroblock subdivisions without severe graphic quality degradation based on object hierarchy information

End of Presentation