Pegasus at the Edge

Workflow Management System

Ryan Tanaka
tanaka@isi.edu
Research Programmer
USC Information Sciences Institute

NSF

USC Viterbi
School of Engineering
Information Sciences Institute

SCITECH

What is the Pegasus WMS?

- Fully featured workflow management system
  - Scientists can develop, run, monitor, debug workflows
  - Utilizes HTCondor as its execution engine
- Jobs can run in shared and non shared file system environments

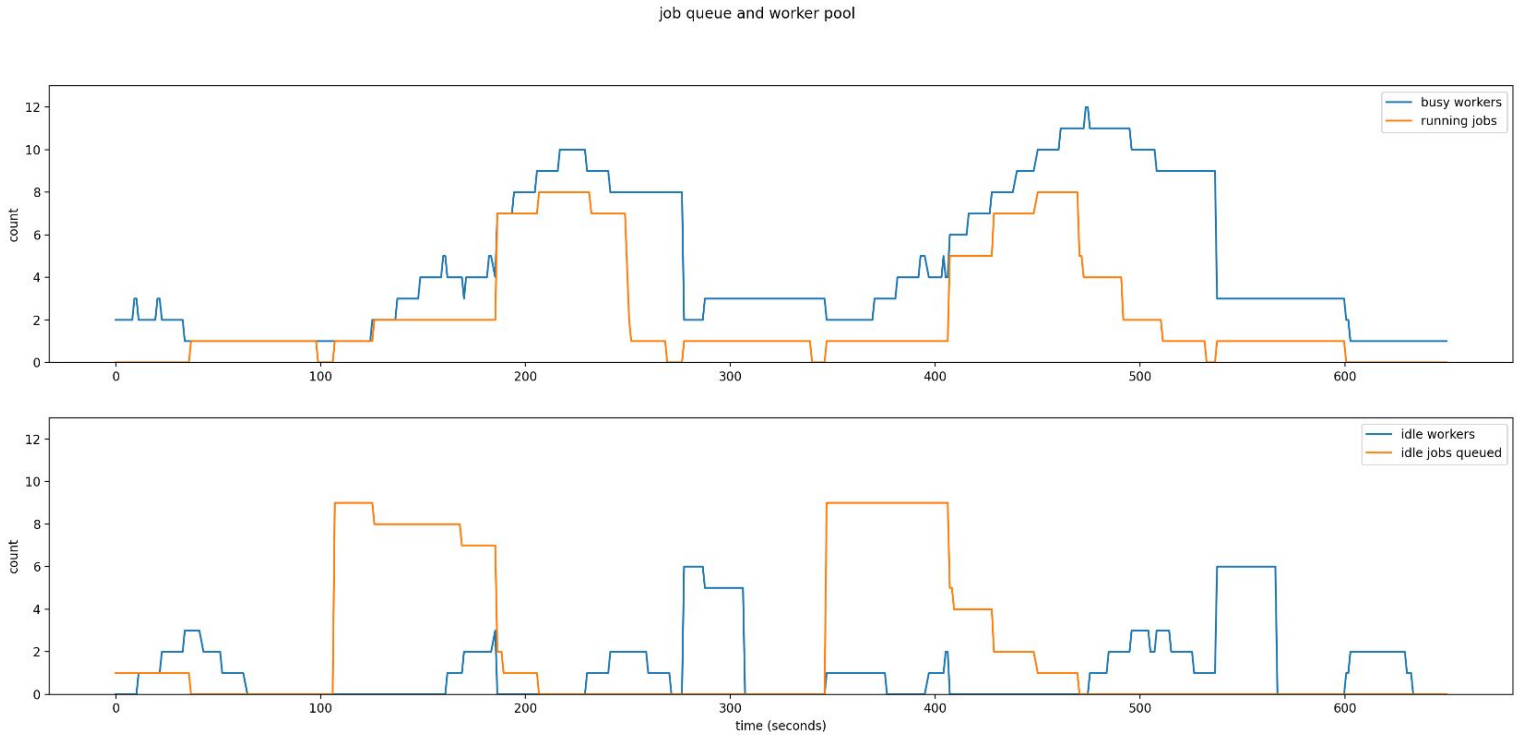
Outline

1. ~~Pegasus WMS~~
2. **Pegasus at the Edge**
3. IoT Hardware and Your Workloads
4. Conclusion

Exploring Pegasus at the Edge: HTCondor Worker Provisioning

- Created HTCondor 9 ARM and X86_64 worker containers
- Built a simple provisioning tool as part of Pegasus
  - If (num idle jobs in queue / num idle workers of same architecture) > load threshold: spin up worker
  - If any worker container sits idle > max_idle_duration: tear down container to free up resources*
    - HTCondor workers are very well optimized and have low overhead, but we will try to free things up when not in use
- Ran experiments locally on X86_64 machine and plan to test on a heterogeneous environment incorporating CHI@Edge resources

Exploring Pegasus at the Edge: HTCondor Worker Provisioning



Outline

1. ~~Pegasus WMS~~
2. ~~Pegasus at the Edge~~
3. **IoT Hardware and Your Workload**
4. Conclusion

Benchmarking FFMPEG on IoT Devices

- Benchmarked FFMPEG performance on RPI4 and Jetson Nano as part of the FlyNet project

Benchmarking FFMPEG on IoT Devices

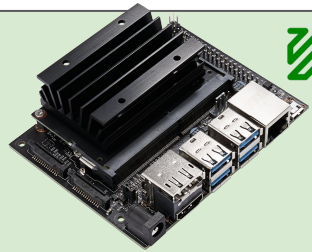
2019 Macbook Pro
4 cores / 8 threads
16 GB ram
SSD with ~ 2400 MB/s write

Latency: ~0.4ms;
Bandwidth: ~ 112 MB/s

Raspberry Pi 4

- Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- 4 GB ram
- Broadcom VideoCore VI GPU
- SD card with ~ 50 MB/s write
- Decode 4K @ 60 | 1080p @ 60 (H.264)
- Encode: 1080p @ 30 (H264)

Jetson Nano



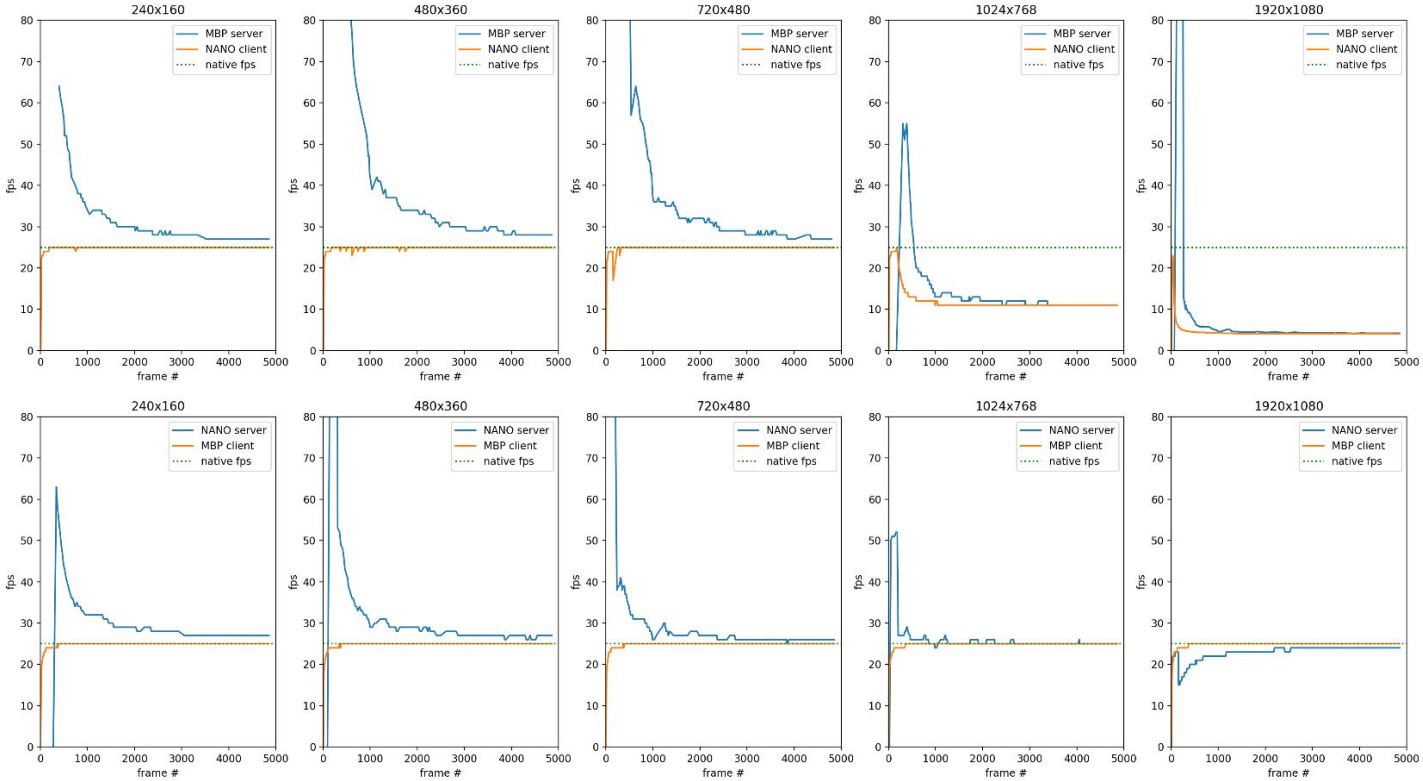
- Quad-core ARM A57 @ 1.43 GHz
- 4 GB ram
- 128-core Maxwell GPU
- SD card with ~ 60 MB/s write
- Decode: 4K @ 60 | 2x 4K @ 30 | 8x 1080p @ 30 | 18x 720p @ 30 (H.264/H.265)
- Encode: 4K @ 30 | 4x 1080p @ 30 | 9x 720p @ 30 (H.264/H.265)

Benchmarking FFMPEG on IoT Devices

- Benchmark
  - 5 identical videos (3 minutes 14 seconds long @ 25fps, h264 encoded) of various resolutions
  - scenarios
    - RPI4 streaming to 2019 MBP client saving individual frames
    - 2019 MBP to RPI4 client saving individual frames
    - Jetson nano streaming to 2019 MBP client saving individual frames
    - 2019 MBP streaming to Jetson nano client saving individual frames

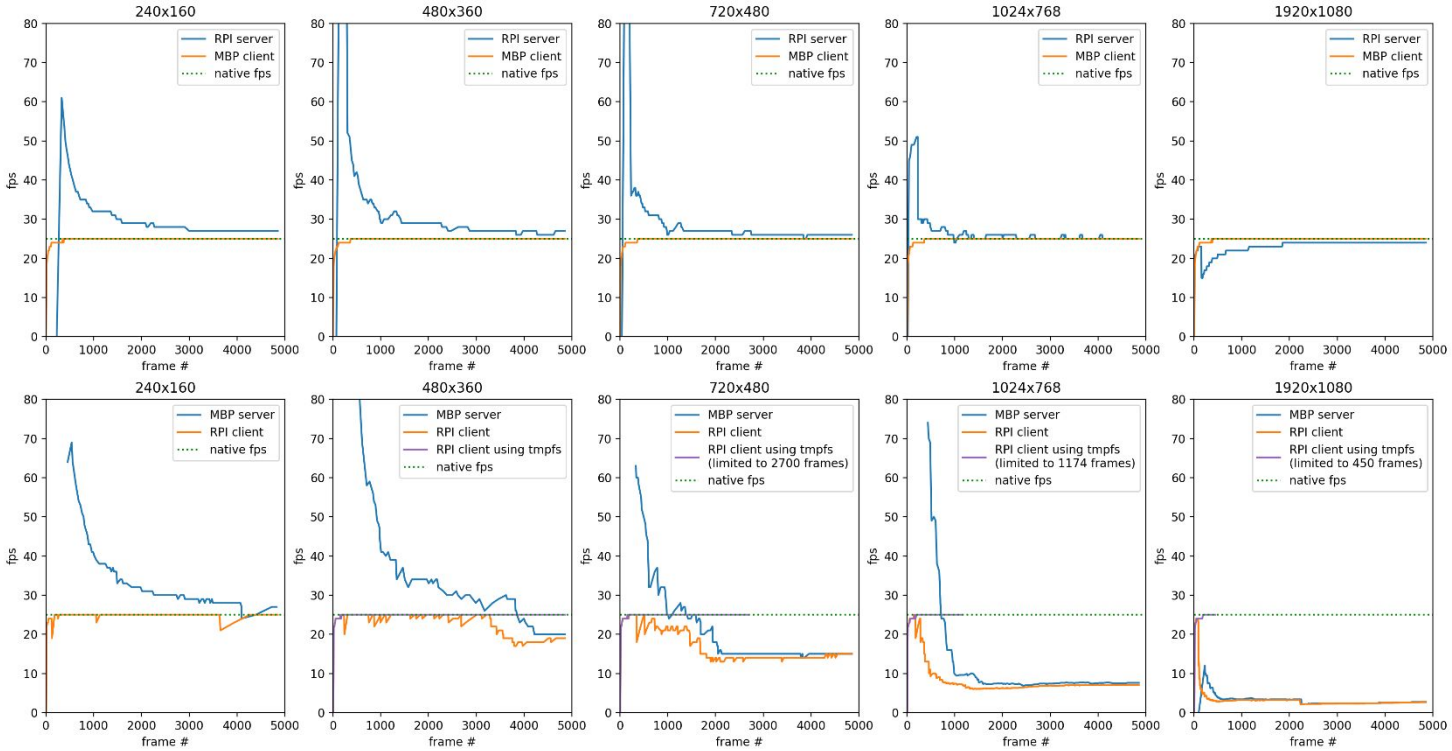
resolution	mp4 size	individual frame size	total size of all frames
240x160	4.7 MB	~ 113 KB	~ 550 MB
480x360	19 MB	~ 507 KB	~ 2.4 GB
720x480	36 MB	~ 1 MB	~ 4.8 GB
1024x768	75 MB	~ 2.3 MB	~ 11 GB
1920x1080	169 MB	~ 6 MB	~ 29 GB

Benchmarking FFMPEG on IoT Devices: Results

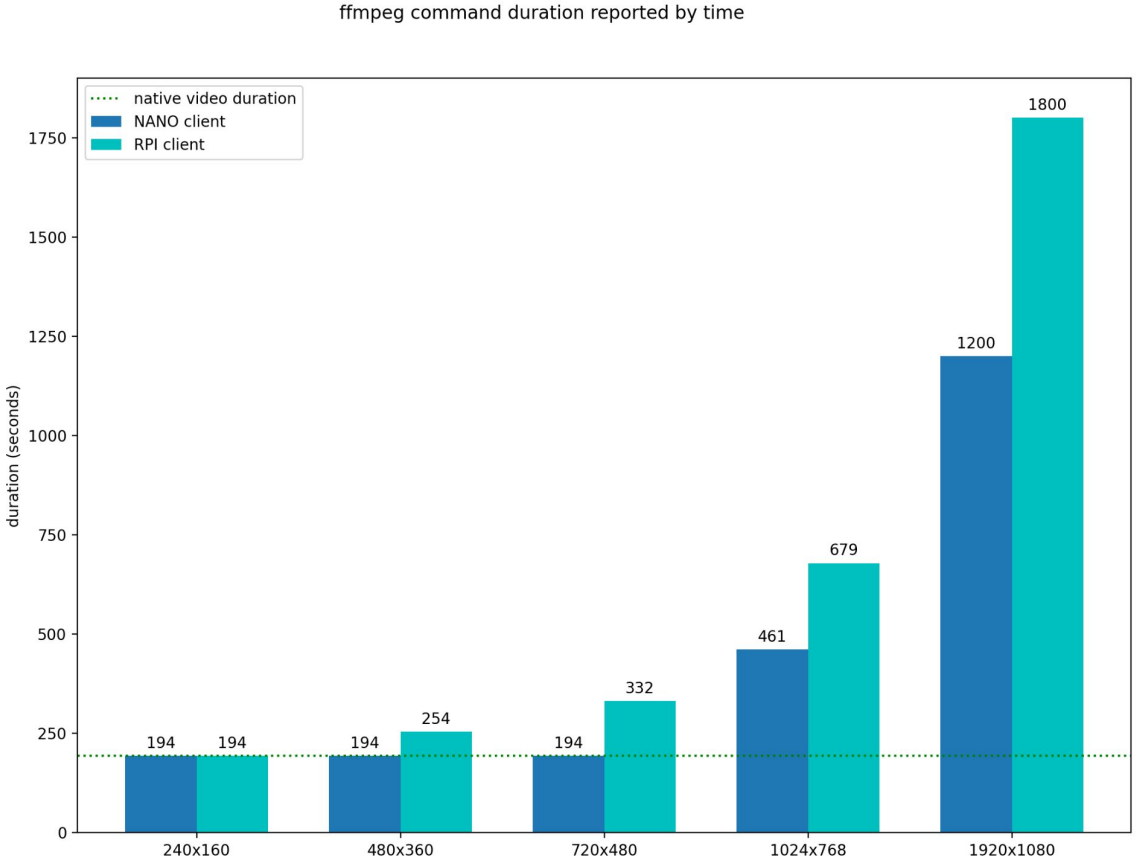


Data obtained by parsing ffmpeg logs on both client and server side.
Each log: "frame= 9 fps=0.0 q=-0.0 size=N/A time=00:00:00.36 bitrate=N/A speed=0.706x"
produced at arbitrary times are recorded to obtain frame (frame # on x axis) and
fps (fps on y axis). This can be interpreted as, "as frame i is being processed,
this is the estimated fps at that time".

Benchmarking FFMPEG on IoT Devices: Results



Benchmarking FFMPEG on IoT Devices: Results



Outline

1. ~~Pegasus WMS~~
2. ~~Pegasus at the Edge~~
3. ~~IoT Hardware and Your Workload~~
4. **Conclusion**

Wrapping Things Up

- CHI@Edge makes it possible for us to test Pegasus on these edge platforms
  - consistent and reliable usage of the python-chi API when interacting with CHI@Edge would be great!
- RPI and Jetson Nano have hardware limitations that can be worked around
  - SD card write speed can be a bottleneck for IO heavy workloads
  - take advantage of main memory when possible

Thank You

USC Viterbi
School of Engineering
Information Sciences Institute