THE INFINITE CAPACITY MEDIA MACHINE

R. I. Cartwright[1] and B. Gilmer[2]

[1] Streampunk Media Ltd., UK and [2] Gilmer & Associates Inc., USA

ABSTRACT

The fast-paced developments of cloud computing are evolving new kinds of machine that will be applied to professional media production. Serverless, distributed, clustered and GPU accelerated, the nature of cloud-fit architectures is changing the patterns and design of software and services. Is it even possible to shift the highly sequential, line-timed infrastructure of existing media facilities into the stochastic, message-driven and asynchronous cloud? What new creative possibilities emerge from a platform that scales to enable personalised production?

The paper imagines the kind of machine you would build if you had an infinite amount of capacity in terms of networks, storage and compute. The design for such a machine and a plan for how to build it are introduced in the form of the *Agile Media Blueprint*. In the context of video data, measurements and analysis are provided for and of the capability of current systems in relation to the idealistic notion of infinite capacity.

INTRODUCTION

Driven by trends such as big data and social media, aggregate performance of information technology is advancing rapidly. Depending on your preferred model (e.g. Moore's Law, Gilder's Law), networking, compute performance and storage capability (bandwidth and capacity) double every six months to three years. In tandem, trends in software development and cloud are reorganising the way that compute resource is configured and accessed, going beyond the boundaries of single computers to exploit massively parallel Graphics Processing Units (GPUs) and compute clusters.

A media company that is planning its future technology platform based on IT trends needs to consider the potential capacity of the machine available at deployment time. Is the rate of performance increase of IT technology outstripping the rate of increase of data rates for media quality, even trends from SD to HD to 4K to 8K with higher frame rates and increased colour depth? If this is the case, consideration should be given to an idealistic machine with the capacity to process as much media data as required, for any given workflow and in a time tending to zero, termed here *the infinite capacity media machine*.

This paper starts by characterising an infinite capacity media machine in terms of media formats, network, compute and storage. Using technologies available today, the *Agile Media Blueprint* [1] - a technical plan for how such a machine could be built today – is introduced. To test how far along the road generic Information Technology (IT) is in delivering the idealist machine, some measurements of media data transported with

commodity networking, processed with GPUs and managed by clustered in-memory caches are presented. The gap between idealistic work and infinite capacity is then analysed, with a description of future work to build the machine and bridge the gap.

INFINITE CAPACITY MEDIA MACHINE

In this section, the various qualities of an infinite capacity machine are examined, from the format of the video to how it is moved, processed and stored.

Format

This paper is specifically about the creation, processing and storage of uncompressed media formats, replacing baseband signals at full quality without trading media quality for bitrate savings. In many current media workflows, compression delivers benefits to overcome transport bottlenecks and storage capacity issues. At the same time, compression adds latency, complexity and increases energy usage. Assuming an infinite amount of capacity, the assertion of this paper is that uncompressed formats are preferable.

Moreover, uncompressed formats have significant benefits in parallel processing environments. Pictures split well into sub-arrays of pixels and each area can be worked on concurrently. Splitting a picture into sequences of consecutive lines moved in tandem has the potential to lower latency during transport. An uncompressed approach saves a lot of packing, unpacking, data duplication and transformation, especially if a common uncompressed format is used throughout the processing pipeline.

That said, when implementing the design for an infinite capacity media machine, compression should be applied where it adds business value. For example, to reduce the long-term storage costs of relatively low value material. The assumption taken by this paper is that is that if you can work uncompressed in the first instance, a cost function can be applied to optimise workflows with compression at a later stage.

Transport infrastructure

Imagine a journey through a media workflow involving moving a sequence of pictures from location A to location B. How you get from A to B depends on the type of transportation:

1. Like a railway, building tracks from A to B and running trains where each carriage is a line of the picture. Only one train can run on the track at any time, requiring timetables, signalling and management, with spare capacity required for resilience. From A to B, a rail network is fast and efficient. However, adding in a new location takes significant planning and cost.
2. Like a road network, where many different routes exist between A and B and each frame or part of a frame is carried in a separate autonomous vehicle. With satellite navigation and live traffic reports, the vehicles each find their own optimal route and, subject to no overall congestion, the parts of the pictures will all arrive in time. Most locations are pre-emptively connected to the road network and resilience comes through redundant routes and timely resending of any missing pieces.

The train network is the equivalent of a dedicated point-to-point media facility, like SDI infrastructure and IP equivalents that use managed networks, such as SMPTE ST 2110.

The road network is like a general-purpose computer network and the optimization of many cloud platforms is in support of Internet infrastructure that is more like roads. Although the railway offers a fast, efficient a reliable form of transport, the road network offers a high degree of agility and personal choice.

If all the pictures arrive in time, does the kind of transport matter? What if the self-optimising road network, where the pictures do not run according to a timetable (e.g. real time), resulted in the content arriving earlier? With infinite capacity, i.e. high-speed roads and no congestion, the agility of the road network is a significant benefit. With 100Gbps non-blocking networking in a local area network, a frame of HD video can be transported at line rate in less than 2 milliseconds, faster if each frame or part of a frame can be transported in parallel. Therefore, this paper assumes self-optimising transport faster than real time, tending to instantaneous, is a benefit to most workflows.

Compute services

Converting media, applying graphics, scaling, mixing, analysing and filtering are a few of the operations that can be applied during a production. In a digital production environment, each of these operations requires compute services. An approach to these services is to split them into composable atomic functions that can be executed on-demand. With an infinite capacity machine, each function is assumed to take zero time to execute.

The massively parallel processing capabilities of modern Graphics Processing Units (GPUs), with many desktop and laptop computers containing hundreds or thousands of computing cores, are a way in which media processing tasks can be split down to a point where the time taken to execute each stage tends to zero. Another alternative is to use compute clusters and functions-as-a-service (e.g. AWS Lambda), with the intriguing possibility of going massively wide, transcoding an entire movie – sub-divided into chunks with each chunk processed concurrently – in a matter of a few seconds.

Storage

Media storage needs to be performant, reliable and scalable to meet the needs of ever-growing volumes of media data. Storage is often a bottleneck in a system, with a choice of video codec being a balance between storage bandwidth / size and the persisted quality. With infinite capacity storage and storage bandwidth, why not store everything at the highest possible quality? Or even at every possible quality? Is media stored just-in-case it is needed again or is the infinite capacity compute capability applied to make each format on demand?

The choice to store media in a format based is an operation that could be carried out based on machine learning.

Many performance issues on websites are resolved with the application of clustered RAM caches. RAM is relatively cheap and many fast turnaround media workflows, such as used in news and sport production, predominantly use a small pool of media clips created within the last 24 hours. An infinite capacity media machine can apply a clustered RAM cache to provide high-performance storage of this pool of media. This can be complemented with longer term storage, such as cloud-based object storage.

AGILE MEDIA BLUEPRINT

The Agile Media Blueprint (AMB) [1] is a development of infinite capacity machine concepts into a technical plan that media companies and suppliers can follow to build highly flexible and scalable systems that run on the same platform as the Internet. By following the plan, media companies are enabled to create and monetize content more effectively than if they were using conventional facilities. The AMB makes use of all the hardware, software, networking and associated components used to run world-wide huge-scale systems such as Twitter.

Overview

The Agile Media Blueprint starts with people, in teams, within organisations and across organisations. As shown in Figure 1, it begins here, not only because people are a key component in our craft-oriented industry, but because by starting here, one can address security at the outset. Administrators can assign roles to people and grant them permissions based upon those roles. People in different organizations may be given specific rights in your organization based upon business needs. For example, personnel from two remote production companies using two different OB vans may be given roles that allow them to collaborate for a one-time event. This approach allows people to securely collaborate using a shared Content API. The Content API can be implemented to interconnect cameras, microphones, speakers, multi-viewers and control surfaces.

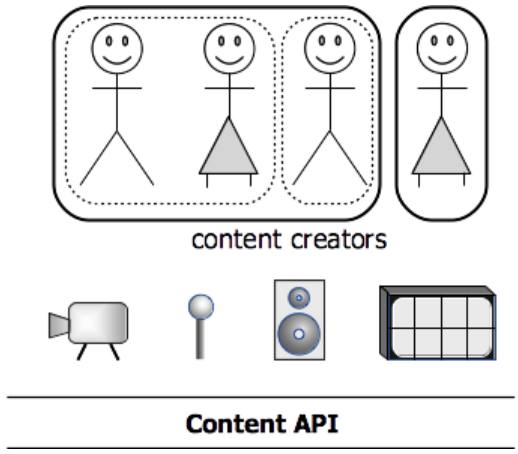

Figure 1 – People & their Roles

While the AMB can be deployed to meet professional media production needs, it also may be used to efficiently produce new types of content that require direct interaction with end viewers. As illustrated in Figure 2, the AMB targets social consumers, through broadcast media, over-the-top streaming (OTT) and mobile.

Conventional broadcast facilities could never provide individualized content to tens of thousands (millions?) of viewers - it is an anathema to the fundamental concept of *broadcasting*. But to deliver whatever a viewer wants to see when they want to see it means that it is critical that the AMB facilities scale seamlessly. In many cases, in an *individualised content* scenario, content is cached and streamed to the viewer as a one-off event. Many broadcasters are already paying for bandwidth at a cost scaled per viewer, without benefiting from a

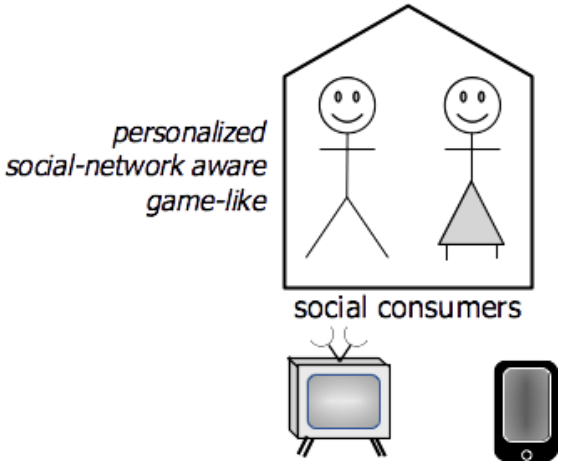

Figure 2 – Consumers interact with bespoke version of content

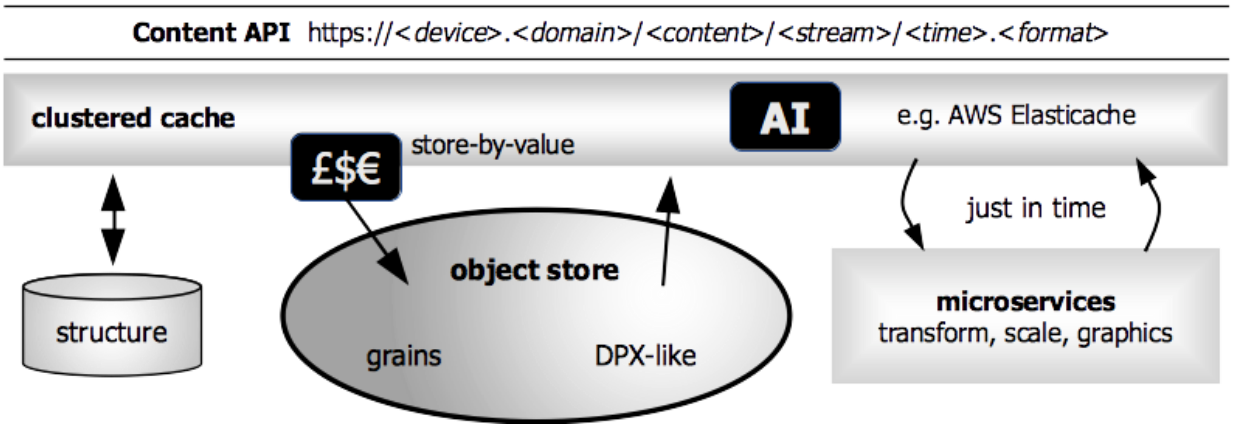

Figure 3 – A common Content API provides a uniform, open interface to essence

closer, bidirectional relationship.

As Figure 3 shows, at the core of the Agile Media Blueprint is a grain-based Content API for element-by-element access to the media, backed by best-of-breed cloud-fit technology, including: clustered RAM caches, AI, object stores. Microservices make new media elements to order as they are required; no more wasting time and resources by creating media elements "just in case" they are needed.

Media transport in the AMB is between Content APIs, with bidirectional links operating in parallel, faster than, slower than, or at real time. As shown in Figure 4, a time-to-bytes translation interface (bytes-to-time component) allows content to be read and written to common signal and file formats, and as described above, allows for compositions to be made on-the-fly.

Libraries of files and streams can be migrated into an implementation of the AMB, either just-in-time, or according to a schedule, using a bytes-to-time unwrap component,

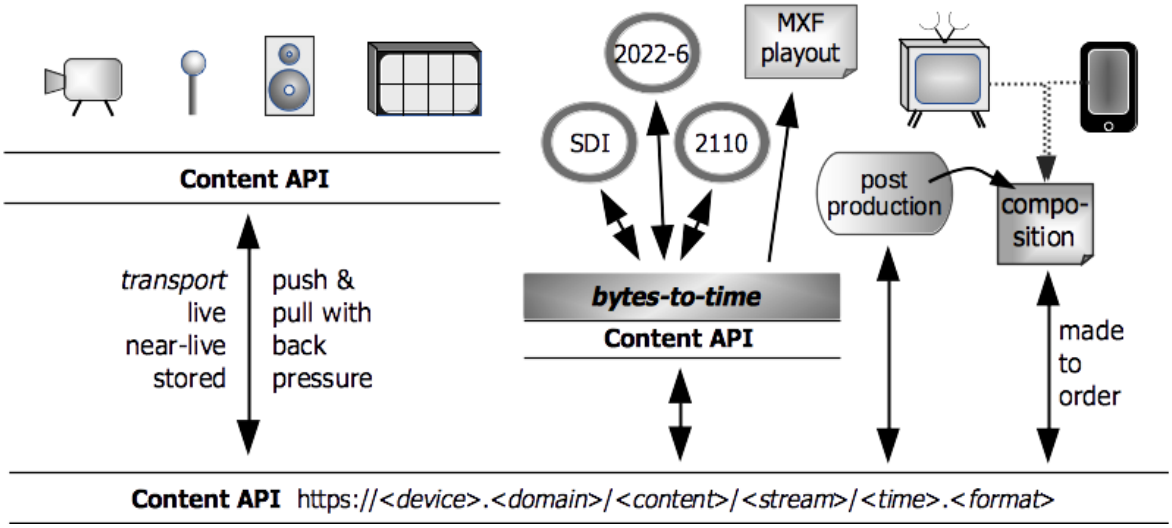

Figure 4 – Bytes-to-time conversion allows creation/consumption of existing formats

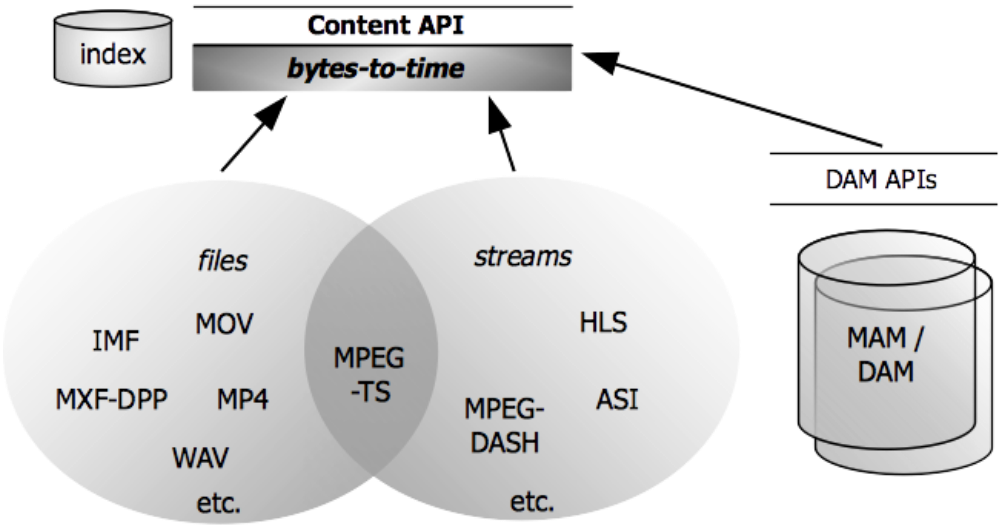

Figure 5 – Bytes-to-time conversion and an index facilitates the creation of common media files and stream formats

supported by an index database (see Figure 5). This allows media factories to easily migrate from a file-based infrastructure to an API-based infrastructure.

MEASURING THE GAP

What is the gap between the idealist infinite capacity machine and current technologies as proposed for the AMB? Is the time taken to process media using the technologies of the AMB tending to zero? This section presents some initial measurements of moving and processing video data on high-speed networks, GPUs and compute clusters. The primary aim of making these measurements is to establish an ongoing process in the industry whereby the capability of the machine for media is routinely benchmarked, informing both strategic and purchasing decisions.

High-speed Networks

Measurements of the speed required to move HD frames of video around a network using HTTP and HTTPS were made as part of testing Internet of Things architectures for cloud-fit professional media [2]. The protocol was chosen because it can be scaled and secured using commodity Internet technology. Results show that in a local area 10Gbps network, it is possible to transport each frame of video faster than real time, with both encryption through TLS and resilience via TCP. Transporting parts of the same streams in parallel is also possible, increasing the maximum throughput.

Parallel Processing with GPUs

Uncompressed frames of video are arrays of millions of pixels. Modern PCs are connected to high resolution screens, HD resolution or better, and are used for applications like video editing, gaming and design. To enable real time rendering of 2D images and 3D environments, dedicated graphics processing chips were developed that can support parallel processing of regions of the picture on hundreds of separate cores. Originally, access to exploit this dedicated acceleration was through graphics manipulation languages

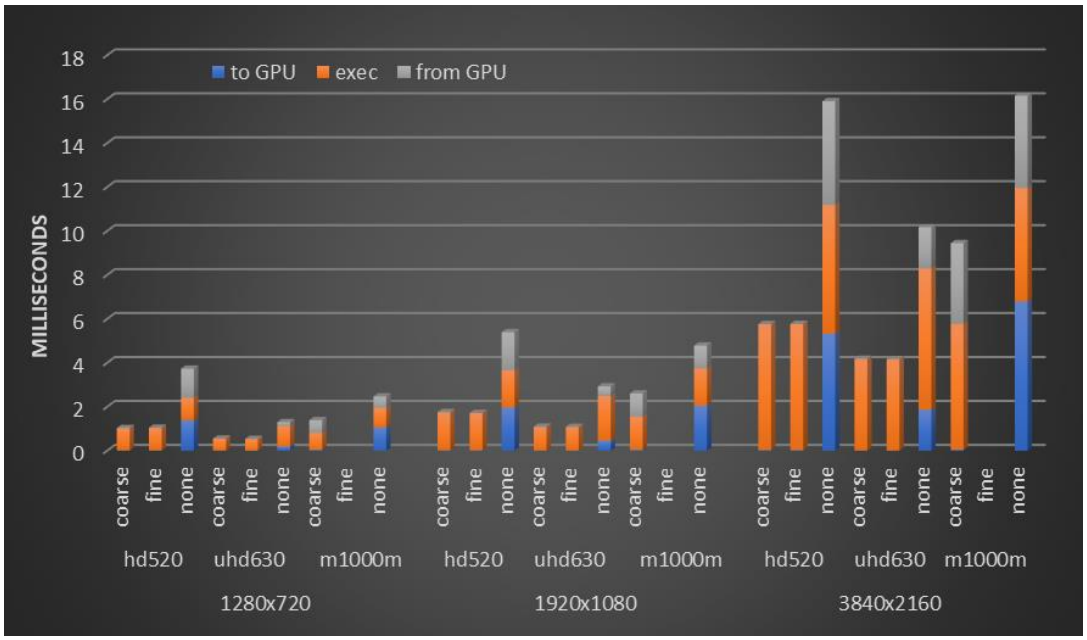

Figure 6 – Measuring GPU media transfer and processing speeds with *nodencl* (https://github.com/streampunk/nodencl)

such as OpenGL and DirectX. In 2009, the OpenCL language [3] was released to facilitate access to the GPU for general processing tasks programmed in a C-like language.

For the processing of uncompressed media, a bottleneck in GPU architectures has been the GPUs dedicated random-access memory (graphics RAM). For media production tasks, each frame of video had to be moved from CPU RAM to GPU RAM, processed, and moved back. The time taken to copy the media and - prior to OpenCL - convert it into a format suitable for processing, is resource intensive and slow. Once on the GPU, media processing pipelines can be constructed in which each stage runs in a few milliseconds or less.

In 2015, Intel introduced their 9th Generation Graphics Architecture [4] in support of the version 2.0 of the OpenCL language. This includes a facility called shared virtual memory (SVM) whereby the CPU and GPU can operate on the same region of memory, either locked off one block at a time (coarse grained) or cooperatively as if the GPU was just another core of the CPU (fine grained). As data does not have to be copied, this means that a frame of video can be handed to and from the CPU to the GPU in a matter of microseconds. When the CPU and GPU are on the same processor die, cache memory is also shared, meaning GPU access to the memory is not limited by the speed of the PCI bus.

The graph in Figure 6 shows measurements of moving uncompressed frames of video to the GPU, performing a simple operation and moving from GPU back to CPU. The results compare times for simulated HD and 4K payloads using data copying (*none*) vs SVM (*fine*/*coarse*) with three different GPUs:

1. Intel Graphics HD520 GPU on same die as an Intel Core i7-6500 CPU
2. Intel Graphics UHD 630 GPU on same die as an Intel Core i3-8100 CPU (~£100)

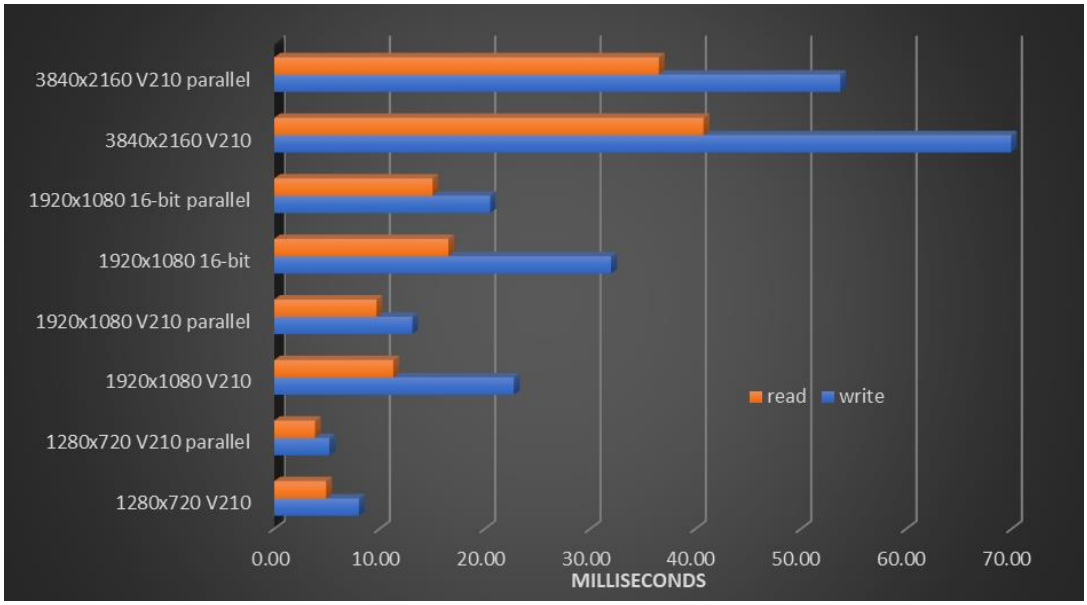

Figure 7 – Measuring Redis cluster speed for simulated video payloads with *redia* (https://github.com/streampunk/redia)

3. nVidia Quadro M1000M GPU connected via PCI to Intel Core i7-6700HQ CPU (fine grain SVM not supported, coarse grain SVM support experimental)

The results show that low cost shared memory architecture can speed up media processing pipelines and carry out complex processing of video images significantly faster than real time, even for uncompressed 4K video running at 120 frames per second.

Clustered Key-Value Cache

To achieve the levels of scale and resilience required by Internet applications, the caching layer of the internet platform has been enhanced to service, manage and orchestrate the processing of requests. By combining fast RAM-based caching with business logic and clustering for scale and resilience, key-value caches (e.g. Memcached, Redis) host much of the backend logic of internet-facing applications. These caches have support for arbitrary data blobs, so what happens if these blobs are media data e.g. video?

The graph in Figure 7 shows basic tests for the read and write times for simulated HD and 4K payloads in and out of a Redis clustered cache, with each operation in series and in parallel. The cache, the smallest possible from the *AWS Elasticache* service, is based on *r4.large* instances interconnected using “up to 10 Gigabit” networking. Collocated in a cloud availability zone with part of the cluster, Node.JS is used as a Redis cluster client running on another r4.*large* instance. For HD media, the frames are moving in and out of the cluster faster than or around real time (allowing for interlace), showing the capability of such a cluster for video processing. For 4K uncompressed video, timings drop below real time. Future work has significant scope to explore faster, larger clusters and optimised client implementations.

Storage

Recent work as part of BBC R&D's cloud-fit project has taken a similar approach for benchmarking media object storage [5], with results demonstrating linear scaling for uncompressed HD video.

ANALYSIS

Infinite capacity is an idealistic notion and, even with sufficient budget, the capacity of the tens of thousands of computers in a cloud data centre could be saturated. For example, providing a unique 4K stream for every one of millions of viewers using uncompressed streams would overload the capacity of such a data centre and likely bankrupt the media company! However, that is an extreme example. For less demanding media factory workflows, where personalisation is achieved by creating compressed media segments that are shared between groups of viewers, or for replicating existing workflows, cloud infrastructure offers orders of magnitude increase in available capacity over a fixed on-site facility of hundreds of servers. In other words, although not *infinite capacity*, apparently limitless capacity is available to explore new creative possibilities.

The definition for an infinite capacity media machine introduced in this paper is the ability to process as much media as a workflow requires in a time tending to zero. Ongoing capacity increase is based on multi-dimensional improvements:

- Agile opportunity - multi-tenancy and on-demand scaling models so that workflows rent just the resource needed when it is required.
- Riding core IT developments – IT technology developments that are a combination of higher-capacity cloud products, cutting-edge computer science and exploiting ever-faster hardware.
- Going wide - the designs of the AMB provide an appropriate framework for executing media workflows using today's clustered cloud infrastructure.
- Optimising software – Developing optimised software that exploits multi-core systems and GPU acceleration for inherently parallelisable media processing.

The measurements presented above of lightly optimised software running on modest instances of the infrastructure show storage, processing and transportation times for uncompressed HD media exceeding real time. This implies that for media factory operations across all dimensions, the processing time is tending to zero.

Should cloud infrastructure be adapted to work more like existing facilities on-premise, e.g. supporting line-based timings with PTP clocks and conformance with the inter-packet arrival times of SMPTE 2110-21? Ideally, yes, to create a common toolkit that can be used from a broadcast truck through to cloud infrastructure. This technology is being requested by media companies from cloud service providers and may be delivered some point soon. However, this approach may be over-engineered as it will couple design choices made for sequential SDI infrastructure (e.g. data rates limited to media real time, every constituent part of every media element processed in sequence) to an otherwise asynchronous cloud machine, effectively limiting full exploitation of compute clusters and/or pipelined GPU processing. Arguably, following this path – taking a constrained model of a legacy facility infrastructure and shifting it to the cloud – will not benefit from all the potential dimensions of increased capacity.

Rather than shifting legacy constraints to the cloud, an alternative is to take an approach that is consistent with modern software development practice. This starts from adopting techniques that hide operational issues such as scaling, resilience and distributed processing behind APIs, then adding in media capability as a thin veneer on top. These API-backed techniques, embodied in clustered or serverless cloud products, were created to provide easy-to-deploy, responsive web applications that dynamically scale to millions of users. Providing business agility while lowering project risk, a media-domain-specific veneer on top can take the form of AMB-defined composable functions as microservices.

Future work

The AMB is a technology plan that needs development through specification, proof-of-concept activities and development and a project plan to build out its components. To date, the AMB has been published as a discussion paper of the Advanced Media Workflow Association intended to facilitate a discussion starting in the autumn of 2018.

Development of open-source Internet of Things framework *dynamorse* [2] is ongoing, including: HTTP/S transport; media encoding and decoding; media transformation – compositing, scaling and mixing; media packing - different bit patterns, colour space translation; file and stream interfaces etc.. The current focus of this work is optimising the framework with memory-efficient OpenCL [3] acceleration of media processing that scales from SD to UHD/WCG content. The tools are deployable via software automation, scaling from RaspberryPi computers, laptops, workstations and up to GPU-accelerated cloud servers. The modules of this IoT framework are candidate first implementations for the microservices of the AMB.

CONCLUSIONS

Strategic planning for the application of IT technology to professional media production could start from a perspective that the machines in use have an infinite capacity, with no processing or transportation latency. Making such an assumption is an enabler for new creative opportunities - such as personalised production - that are enabled through the scale and multi-dimensional innovations of cloud computing.

Rather than trying to shift current facility architectures to the cloud, systems can be designed to get as close as possible to the idealistic infinite capacity machine, with the Agile Media Blueprint providing a plan for how to achieve this. Taking this approach will result in alternative workflow realisations that make use of compute clusters and serverless architectures, benefitting from agile, more performant, parallel and distributed computer technology. Benchmarking speed and modelling the cost of deploying such systems for media processing should become an ongoing activity of any media project. By using the same infrastructure that scales to deliver over the top (OTT) services and thinking differently about how to deliver media workflows, apparently limitless orders of magnitude increase in media machine capacity over current facilities are available.

REFERENCES

1. Cartwright, R. I. and Gilmer, B. 2018. Agile Media Blueprint – creating and monetizing content using the Internet technology platform. Discussion paper of the Advanced Media Workflow Association. Available: https://www.amwa.tv/downloads/reference_documents/The_Agile_Media_Blueprint.pdf

2. Cartwright, R. I. 2018. An Internet of Things architecture for cloud-fit professional media workflow. SMPTE Motion Imaging Journal. June 2018. 3. The Khronos Group. 2009. The Open CL Specification. Version 1.0, rev 48 and subsequent versions. Khronos Group specification. Available: https://www.khronos.org/registry/OpenCL/

4. Junkins, S. 2015. The Compute Architecture of Intel Processor Graphics Gen9. Version 1.0. Intel whitepaper. Available: https://software.intel.com/sites/default/files/managed/c5/9a/The-Compute-Architecture-of-Intel-Processor-Graphics-Gen9-v1d0.pdf

5. S. Nicholson, 2018. Beyond Streams and Files – Storing Frames in the Cloud. BBC R&D blog. Available: https://www.bbc.co.uk/rd/blog/2018-03-cloud-video-production-stream-file-frame

ACKNOWLEDGEMENTS

The authors would like to thank the IBC conference committee for permission to publish this paper. Thanks also to: the team at the BBC in Northern Ireland for providing equipment used as part of this testing and support for the development of the AMB; to the AMWA for publishing the AMB discussion paper; Simon Rogers for his technical advice and support.