Help Save The Planet: Please Do Adjust Your Picture

Oche Ejembi
School of Computer Science
University of St Andrews, UK
ooe@st-andrews.ac.uk

Saleem N. Bhatti
School of Computer Science
University of St Andrews, UK
saleem@st-andrews.ac.uk

ABSTRACT

Allowing digital video users to make choices of picture size and codec would significantly reduce energy usage, electricity costs and the carbon footprint of Internet users. Our empirical investigation shows a difference of up to a factor of 3 in energy usage for video decoding using different codecs at the same picture size and bitrate, on a desktop client system. With video traffic already responsible for the largest and fastest growing proportion of traffic on the Internet, a significant amount of energy, money and carbon output is due to video. We present a simple methodology and metrics that can be used to give an intuitive, quantitative and comparable assessment of the energy usage of video decoding. Providing energy usage information to users would empower them to make sensible choices. We demonstrate how small energy savings for individual client systems could give significant energy savings when considered at a global scale.

1. INTRODUCTION

Video has become a popular use-case for many of today's Internet users. With 65% of all Internet traffic being some form of video [1], it is already the largest and fastest growing class of traffic today. This percentage is expected to rise to as much as $\sim 80\%$ – $\sim 90\%$ by 2017, as new technology makes it easier, cheaper and more accessible to send and receive video. While it is still most popularly used for entertainment / leisure purposes (e.g YouTube, Netflix, Vine etc.) and communication purposes (e.g Skype, Facetime, etc.), Internet video is also used in several other crucial, important applications such as education [2], surveillance [3], healthcare [4], and agriculture [5].

With this growth in its usage, there must be a commensurate growth in its energy consumption and carbon footprint. Considering that the developing regions of the world e.g. parts of Asia, Africa and South America, have very large populations and already contribute significantly to the global number of Internet users, but still have relatively low Internet penetration (typically well below 50%), the energy

load of video could be extremely significant. Indeed, video is a great enabler for countries where functional literacy is low: video communication has greater utility for users than text-based communication. It has been conservatively estimated that in 2014, ICT would contribute 3.2% of all carbon emissions globally, greater than that produced by the aviation sector [6], and so video will be responsible for a significant proportion of this.

1.1 Video diversity

Today, there are many video codecs that are widely used. The diversity of video codecs also provides an opportunity for users to exercise choice in their usage. There are several trade-offs involved in such a decision. Some codecs might offer a wider range of options and configurability than others, while some perform best for particular use-cases and scenarios. Consequently, there are several metrics for Quality of Service (QoS) and Quality of Experience (QoE) involved in benchmarking and selection of codecs. While energy usage is of some concern to users, especially for mobile devices which have a constrained energy supply in the form of battery, there are currently no well-defined metrics which specifically take energy usage of codecs into consideration in a way that can be (i) integrated into applications, and (ii) be understood by users so that they can be empowered to make appropriate choices.

1.2 Contribution

We have investigated client-side energy usage of a number of popular and important video codecs: Flash video (Sorenson Spark/FLV1); MPEG-4 Part 2 (two variants); MPEG-4 Part 10 / H.264; H.265 / High Efficiency Video Codec (HEVC); and Google's VP8 & VP9. We have used a simple testbed to examine the energy and compute-resource usage of these codecs. Our contributions are:

1. We define a simple methodology and metrics to provide intuitive, quantitative and comparable energy usage and quality assessment for video codecs.
2. Applying our methodology in an inexpensive desktop testbed, we present empirical measurements and assessment of significant differences in energy usage across the codecs.
3. We propose ways in which the energy metrics can be integrated into video applications and how users can be empowered to make choices about codec usage based on measured energy usage information.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MM'14, November 3–7, 2014, Orlando, Florida, USA.

Copyright is held by the owner/author(s). Publication rights licensed to ACM.

ACM 978-1-4503-3063-3/14/11 ...\$15.00.

<http://dx.doi.org/10.1145/2647868.2654897>.

We focus here on decoding (playback) of pre-existing video content, as video decoding events today far outnumber video encoding (creation) events, e.g. consider video downloads from YouTube. However, YouTube is also a prime example of how video creation is becoming widespread, so we also include a discussion of how our methodology and findings would apply to encoding of video. We also consider some hardware issues for video decoding, but as most of the video codecs available to users are software only, the energy savings we present are complimentary to energy-efficient hardware, can be applied to legacy systems that are not ‘green’, and so have the potential to offer huge energy savings *globally*, from relatively small *individual* client-side and server-side savings, especially as use of ICT grows.

2. RELATED WORK

In recent years, studies in energy awareness for ICT systems (or ‘Green IT’) have gained increasing interest, both in research and industry. While most of these studies have focused either on hardware or data-centre optimisations, some have investigated how software behaviour can have significant impacts on energy usage. For example, Sabharwal et. al. [7] discuss several aspects of software architecture which can affect energy use. Based on testbed measurements, they show how relatively simple optimisations in source code can lead to considerable energy savings (e.g. improved laptop battery life).

The human users of these software applications also have an important role to play in energy-efficiency. In our previous work [8], we show through a 2-year study, how providing feedback and incentives to IT-system users encouraged behavioural changes towards saving energy. We recorded savings of up to 56% in individual energy consumption. Similarly, Amsel et. al. [9] present a tool which provides measurement-based feedback on application energy usage to users, who may then modify their behaviour accordingly.

With the widespread use of video in ICT, previous work has looked into making video ‘greener’. Again, while the bulk of the work has looked at improving hardware, some have looked at software based optimisations at various levels. Li et.al. [10] examine improving energy efficiency at the application layer using mechanisms such as dynamic cache management. Nencioni et. al. [11] present a predictive prefetching algorithm to save energy used by the BBC iPlayer TV-catchup service. Song et. al. [12] examine optimising the energy use of video transcoding CPUs using a modified DVFS algorithm.

Video diversity, as discussed in Section 1.1 above, has also generated considerable research interest. Several works, e.g. [13–15], have compared various video codecs for typical quality of experience (QoE) metrics, such as peak signal to noise ratio (PSNR), structural similarity index metric (SSIM) and mean opinion scores (MOS). Also quality of service (QoS) metrics, such as network bit-rate variability and bit-error or delay tolerance, have been considered. Other works have investigated the characteristics, performance and resource usage of various popular video *applications* such as YouTube, Netflix and Vine e.g. [16–18]

Surprisingly, very little work has explicitly examined the energy use of different codecs as used in these popular applications. However, Lin et.al [19] present a (now dated) investigation on the energy use of some codecs on a mobile platform. Sharrah et. al. [20] present an analytical model

of the power consumption of capturing and *encoding* using H.264 and MPEG4 codecs.

We have investigated how choice of codec and picture size for video playback impact client-side energy usage. This is the first work to investigate the energy consumption of the newest codecs, VP9 and H.265, which might increase in popularity, and compare them with popular codecs including H.264, VP8, Flash Video (FLV) and MPEG-4 Part 2. This is also the first work to propose providing information to users of video energy usage.

3. EXPERIMENT METHODOLOGY

The aim of our experiment was to perform an empirical investigation of the energy and resource consumption of various popular codecs during encoding and decoding (playback). Based on that energy usage, we have defined how energy usage information can be presented to users to enable choice of codec to be made.

3.1 Overview

We chose two opensource, high-definition (HD) video files, as summarised in Table 1. These were then re-encoded at different picture sizes, and with several different codecs, using the opensource FFmpeg¹ software. The picture sizes we chose cover a range of common use-cases, as described in Table 2. The video codecs are a mix of some that are commonly used and some that might become popular, and are listed in Table 3. Only default settings were used for encoding the test video files, whilst varying only the picture size of the output video and target bit-rate to obtain similar file attributes.

Table 1: Opensource video files used in experiments.

Big Buck Bunny: A popular <i>animation</i> video, featured in a number of published experiments, e.g [21]. We used the first 2 minutes of this video.
Tears of Steel: A science-fiction, <i>CGI-enhanced, live-action</i> movie. We used 2 minutes starting from the 6:00 mark, containing fast-paced action scenes.

Both files, from Blender (<http://www.blender.org> – an opensource movie project) were retrieved from an open, public repository (<http://www.media.xiph.org>) in YUV format.

Table 2: Video picture sizes and example use cases.

Label	Picture size	Target bit rate (Kbps)
QCIF ^a	176x144	128 (Older handheld devices)
CIF ^b	352x288	256 (Handheld mobile devices)
360p	480x360	640 (Smartphones)
480p	720x480	728 (Smartphones, tablets)
720p	1080x720	1456 (HD, desktops, TV)
1080p	1920x1080	2912 (‘full’ HD, desktops, TV)

^aAlso known as 144p. ^bAlso known as 288p.

3.2 Testbed

The testbed is shown Figure 1. A single host was used to decode and playback the encoded video files – two of these units are shown in the left of the figure. Each system is a Shuttle XPC Glamor SG31G2², with Intel® Core™ 2 Quad

¹<http://ffmpeg.org>

²<http://global.shuttle.com/unique/productsSpec?productId=646>

Table 3: Video codecs: libraries and software.

Label	Library & software version
FLV	from ffmpeg-libavcodec v55.52
H264	libx264 v0.125
H265	libx265 v0.8
MPEG4	from ffmpeg-libavcodec v55.52
MSMPEG4	from ffmpeg-libavcodec v55.52
VP8	libvpx VP8 v1.3.0-1780-g1f08824 ^a
VP9	libvpx VP9 v1.3.0-1780-g1f08824 ^a

^aFrom the WebM Project <http://www.webmproject.org/>

Q6600, 2.40GHz CPU with a Intel 82G33/G31 chipset and an Express Integrated Graphics Controller core, 4GB RAM (128MB used for graphics, set in BIOS). To examine the effects of hardware-assisted video decoding, we performed experiments with a PCI-express Nvidia® G86 GeForce™ 8400GS with 256MB GDRAM. This is considered a mid-range desktop configuration. The machine ran a minimal installation of Ubuntu Linux 13.10 64-bit (x86-64) Server installation, with no desktop environment, and with a minimal set of background processes running. This was to avoid any unnecessary load on the machines when measuring energy usage. However, to enable video playback on the client machine, we used the lightweight *Openbox*³ window manager. We played-back the video files using the opensource VLC media player application, version 2.2.git Weatherwax (revision 2.1.0-git-2995-dg136375)⁴.

The machine used for encoding the video files from the YUV source was a mini-tower workstation (on the right in Figure 1) with an Intel Core i5 4440, 3.1GHz, Quad-Core (Haswell) processor, 4GB (2x2GB) Corsair DDR3 (1333MHz) memory, and a C220 Series Chipset. This ran the Ubuntu Linux 13.10 64-bit (x86-64) Desktop installation. This is considered a mid-range workstation configuration, which is affordable for home use. No hardware assist was available for encoding video and so this was not assessed. This is fairly typical at the time of writing, though hardware assisted encoding is becoming available, and we would expect this to be more common in the future.

Figure 1: Desktop testbed. The black Shuttle systems on the left were used for the decoding experiments, and are connected to the Kill-A-Watt/MoteWatt power meters in the centre. The mini-tower on the right is the workstation used for encoding the videos from the YUV HD source.

We obtained power readings for the experiments using our modified Prodigit ‘Kill-a-Watt’ 2000MU power meter⁵,

³<http://openbox.org>

⁴<http://www.videolan.org>

⁵<http://www.p3international.com/products/p4400.html>

dubbed the ‘WattMote’ (the two units in the centre). The unmodified meter is a cheap but sensitive and accurate real-time home power meter. Using instructions available online [22], the meter was modified with an RFM12B radio to enable captured readings to be sent to a receiver (one of the other Shuttle units, on the left).

3.3 Measurement regime and observables

We took power measurements for decoding and encoding for each combination of the 7 codecs (*vvv*) and 6 picture sizes (*ppp*), a total of 42 combinations. The measurement was for the system as a whole, as this is the power usage seen by the user when undertaking the {de,en}encoding tasks. Each power measurement was repeated 20 times for decoding, and 5 times for encoding, which we found to give acceptable variations for either experiment. As each video sample was 2 minutes long, for the 2 films this is a total of 3360 minutes of measurements for decoding. Encoding times varied greatly, depending on codec and picture size, from 0.5 seconds to 360 minutes.

Each *vvv* – *ppp* combination used only the software codecs and the main system CPU. The NVIDIA graphics card supports hardware assist for H.264 decoding using libVDPau v304.88, so a separate set of measurements were conducted for H.264 with hardware assist – 6 picture sizes, 20 runs, 2 minutes per run for each film, for an additional 480 minutes of tests. We also measured the system power usage when idle to assess the additional power usage of the systems as a whole due to the {de,en}coding tasks.

The observables we capture for this experiment are summarized in Table 4. The derived metrics we used are explained in Section 4. Our results are detailed in Section 5.

Table 4: Summary of observables for experiments

Observable	Units	Measured using
Power	Watts	MoteWatt power meter
{De,En}coding Time	Seconds (s)	<i>time(1)</i> ^a
Energy	K Joules (KJ)	Power × time for {de,en}coding
CPU usage	Percentage	<i>time(1)</i> ^a
RAM usage	MiB	<i>time(1)</i> ^a
PSNR	dB	<i>tiny_ssim</i> ^b
SSIM	–	<i>tiny_ssim</i> ^b

^aNote that this is `/usr/bin/time` and not the *time* command that is built into the default *bash* shell on linux.

^bhttp://ffmpeg.org/doxygen/trunk/tiny_ssim_8c_source.html

4. ENERGY AND QUALITY METRICS

We describe the metrics we have used to (i) evaluate our measurement-based study; and (ii) demonstrate how energy information could be used within video applications.

4.1 Energy usage for video

For video, we are concerned with energy usage with respect to the video stream, so we choose to define a metric which measures *energy used per second of video stream content*. Such a definition is independent of the way the video is encoded, and so can be used to make comparisons across

different codecs. It is also a simple metric for users and developers to understand. Note that the units of our metric, which we will call P_{xv} (where $x = d$ for video *decoding* and $x = e$ for video *encoding*), are then *Joules per second (of video content)*, i.e. J/s_v . These are the same units as electrical power (Watts, $W = J/s$), but we choose to write the units as J/s_v to make clear the different assessment we are making. However, this does mean that we can also translate directly from measured power usage for a system to our video energy metric P_{xv} , which is very convenient.

$$P_{xv} = \frac{\text{energy usage for video \{en,de\}coding}}{t_v} = P_a - P_{idle} \quad (1)$$

where P_{xv} is the mean energy usage per second of video (J/s_v) by video encoding (P_{ev}) or decoding (P_{dv}), and t_v is the duration of the video stream in seconds. P_a is the mean measured power of the system during encoding or decoding of the video. P_{idle} is the mean measured power when the system lies idle. This is summarised in Table 5.

Table 5: Definitions for P_{xv} (Eqn. (1)).

P_{xv}	mean energy usage per second during video encoding($x=e$) or decoding($x=d$) [J/s_v]
P_a	measured mean <i>system</i> power usage during video encoding or decoding [W]
P_{idle}	mean system power usage when idle [W]

4.2 Energy metric

We wish to assess the amount of energy used for a system (workstation or server) as a whole in either encoding or decoding video. It would also be possible to assess the energy usage for individual components of a system, including hardware components (e.g. CPU memory, graphics card) and software (e.g. operating system, graphics driver, etc). For now, our concern is with energy usage as a whole, but more detailed analyses of energy usage of individual components may yield benefits for targeting or optimising energy efficient operations – we leave this for future work.

4.3 Decoding video

We wish to create a normalised decoding metric, E_{dv} . We define $_{vvv}E_{dv-ppp}$ as the energy metric for decoding video using codec $_{vvv}$ (from Table 3) at picture size ppp (from Table 2). We define $_{vvv}E_{dv-ppp}$ for decoding video:

$$_{vvv}E_{dv-ppp} = \frac{P_{idle}}{P_a} \quad (2)$$

$$= \frac{P_{idle}}{_{vvv}P_{dv-ppp} + P_{idle}} \quad (3)$$

with P_a , P_{dv} and P_{idle} as defined in Section 4.1. Note that the denominator is simply the mean power measured during the decoding process. For example, for decoding using H.264 to 720p, we would define:

$$_{h264}E_{dv-720p} = \frac{P_{idle}}{_{h.264}P_{dv-720p} + P_{idle}} \quad (4)$$

4.4 Encoding video

For our experiments here, we chose to use the measured energy required to convert the full-quality source, which is a YUV encoding of picture size 1920x1080 and 30fps, to the encoding scheme supported by the video codec while maintaining picture size and frame rate. So, using similar notation to that for decoding, we define E_{ev} :

$$_{vvv}E_{ev-ppp} = \frac{P_{idle}}{_{vvv}P_{ev-ppp} + P_{idle}} \quad (5)$$

where $_{vvv}$, ppp and P_{idle} are as defined for the complementary decoding expression given in Eqn. 3. For example, for encoding using H.264 at 720p, we would define:

$$_{h264}E_{ev-720p} = \frac{P_{idle}}{_{h.264}P_{ev-720p} + P_{idle}} \quad (6)$$

4.5 Limitations of E_{xv}

From the discussion above, the metric, E_{xv} is defined in terms of real energy measurements. This has the advantage that, being defined in terms of P_{xv} from Eqn. (1), values are comparable across the system for different codecs. Creating a purely analytic model for evaluating an energy metric might be more convenient from a methodical viewpoint, but it is not practical: it would need to encompass the complexities of the different video encoding/decoding algorithms, the specific implementations of those algorithms, and the range of hardware and software components that are involved. Our measurement-based approach removes that complexity by treating the system as a black box.

However, one drawback of a measurement-based approach is that all measurements are system-specific: the same methods used to conduct the same measurements on a different system would yield different results. Differences in implementation of the same video encoding and decoding algorithms, as well as differences in hardware and software components and configuration mean that the actual values of $_{vvv}P_{dv-ppp}$ and $_{vvv}P_{ev-ppp}$ could be very different for the same coding scheme ($_{vvv}$) when measured on different systems (or even on the same system but with different configurations of codecs, drivers, etc). Additionally, some systems might have hardware assist for specific algorithms, reducing their values of $_{vvv}P_{dv-ppp}$ and $_{vvv}P_{ev-ppp}$, where as other systems might use only software. For example, many modern smartphones will have hardware assist for H.264, but many low-end desktop machines may not. However, where a set of measurements are made relative to the same system, the values of $_{vvv}P_{dv-ppp}$ and $_{vvv}P_{ev-ppp}$ do provide indicative energy performance comparisons, and additional measurements can be made to cater for hardware assist for direct comparison also. Indeed, this is what we have done in our measurements.

4.6 Combined quality-energy metric

To demonstrate how the energy values could be incorporated into an application, we take a simple approach, using a compound metric as in Eqn. (7) and Table 6.

$$Q_M = \sum_{m=1}^M \alpha_m Q_m \quad (7)$$

$$0 \leq Q_m \leq 1 \quad (8)$$

$$\sum_{m=1}^M \alpha_m = 1, \quad 0 \leq \alpha_m \leq 1 \quad (9)$$

Table 6: Definitions for Q_M (Eqn. 7).

Q_m	an application quality metric, e.g. for a video codec or some other relevant quality metric
α_m	a weight

With respect to Eqn. (7), E_{xv} (from Eqn. 3 or 5) would be treated as a quality measure. This is very convenient to incorporate into applications, allowing application control and decision policies to use energy along with traditional quality measures. Additionally, if more than one video quality metric was required for assessment of quality, then those multiple metrics could also be integrated as different Q_m , using appropriate weights α_m .

However, if such a method of composing the energy metric into an applications is not suitable, then a more appropriate algorithm could be used, as required. The intention of our presentation here is to demonstrate that from a measurement-based approach, it is possible to derive an energy metric that could be incorporated into applications.

4.7 Examples of codec quality metric Q_m

There has been much debate within the community about which objective video metrics are appropriate for measuring video quality, e.g. [23]. Rather than take a position in that discussion, our quality assessment leaves the choice of metric to the user and only requires that it conform to the normalised range and semantic as given in expression (8). Existing quality metrics should be transformed in order to be used as we have expressed in Eqn. (7).

For example with SSIM [24], the value is already normalised to the range [0,1]. However, PSNR must be converted in a way that makes sense to the applications. One method for converting PSNR is to note that some studies have observed a near-linear correspondence between PSNR and MOS [25], allowing normalisation and mapping to the appropriate range for Q_m as given in expression (8).

5. RESULTS

Our focus is on our decoding results, but we also consider briefly some of our encoding experiments.

5.1 Decoding measurements

Figure 2 shows the results from the decoding experiments entirely by software using the CPU without the hardware card installed. The H.265 codec, overall, consumes the greatest energy, followed by VP9. H.264 and VP8 consume an equal amount of energy and system resources during playback, while FLV consumes the least amount of resources during playback. It is important to note that these differences in energy and resource usage are at roughly the same target bit-rates. There are differences in the perceived quality of the decoded file, which we capture using SSIM and

PSNR values in Figures 4. While the use of such metrics (especially PSNR) remain in debate, they do give indications of the level of perceived video quality (especially the SSIM metric), and help us to explain how video quality and energy usage can be used together (Section 6). More suitable metrics could be substituted as requirements dictate.

5.2 Limitations of QoE measurements

The results of the PSNR and SSIM assessments are shown in Figure 4. We observed that the popular opensource x264 codec (considered to be the best implementation of the H.264 codec standard [14]), produces poor results when analysed by the tiny_ssim tool [26]. However, a quick visual examination of the resultant videos shows that the quality is good, comparable to VP8, VP9 and H.265, and visibly better than MPEG4 and FLV. A possible explanation is that x264's default psycho-visual optimisations provide improved visual quality to human viewers, but negatively impact calculated PSNR and SSIM scores. Another limitation is in how tiny_ssim is implemented. It requires that the comparison it makes is on two YUV files, so we need to transcode our encoded video files to YUV, e.g. the H.264 video file is converted to YUV format before being used by tiny_ssim, and some of x264's optimisations may not be visible in the YUV format.

We used tiny_ssim as it allowed us to assess all the codecs. Other tools (e.g. the widely-used MSU Video Quality Measurement tool [14]), do not support H.265, VP8 and VP9.

6. TRADE-OFFS AND SAVINGS

We focus on discussing decoding, for the sake of brevity, but similar analysis is possible for encoding also.

6.1 Software only decoding

Values for metrics for P_{dv} , E_{dv} , and Q_{dSSIM} are shown in Fig. 6. P_{dv} is as from Eqn. 1, and the graph shows the energy consumption of video playback. We observe very small differences among the various codecs for picture sizes up to 360p. For example, at 360p (which is YouTube's default size) all but one of the codecs consume approximately $4-5J/s_v$. H.265 consumes the most energy at this picture size, $\sim 7J/s_v$. From 480p, we begin to notice significant differences among the codecs. At 1080p we see a difference of factor of 3 between FLV (lowest P_{dv}) at $15J/s_v$ (3 times more than at 360p), and H.265 (highest P_{dv}) at $53J/s_v$ for 'Tears of Steel' (~ 8 times more than at 360p).

The E_{dv} graphs show the normalized values of P_{dv} as calculated using Eqn. (5). These values for E_{dv} are used to create Q_{dSSIM} as in Eqn. 7, along with measured SSIM values to represent quality, each with equal weight. For Q_{dSSIM} , we find that at QCIF picture size, the VP9 codec provides the best energy-quality tradeoff for the 'Big Buck Bunny' movie, whereas for 'Tears of Steel', it is the HEVC codec. MPEG4 and H.264 rank lowest at this picture size, because for approximately the same amount of energy, they provide a relatively poor quality. At 360p, the MSPEG4 codec provides the best energy-quality trade-off, while at 1080p it is FLV (for 'Tears of Steel'). However, please note our comments in Section 5.2 on our use of tiny_ssim.

We have observed that at different picture sizes, different codecs may provide the best trade-off between energy use and picture quality. Based on this, an energy-aware application could dynamically adapt its picture-size and/or video

Figure 2: Resource usage for decoding of video files. 20 runs were used to plot each point, with 95% confidence (error bars not always visible as they are very small in some cases). The lines joining the points are visual aids only and do not imply interpolation to intermediate picture sizes.

Figure 3: Resource usage during encoding video files. 5 runs were used to plot each point, with 95% confidence (error bars not always visible as they are very small in some cases). The lines joining the points are visual aids only and do not imply interpolation to intermediate picture sizes.

codec with the aim of conserving energy while maintaining the best possible quality [27, 28]. (A similar analysis is possible for the encoding results.)

6.2 Hardware assisted decoding

Our NVIDIA G86 (Geforce 8400) graphics card supports only decode for H.264 from our chosen codecs. However,

we can still draw some general conclusions. We compared the energy use of the system using hardware and software and present results in Fig. 5. Using hardware can make the playback process more energy efficient. At 360p, while software decoding consumes $\sim 5J/s_v$, hardware-assisted decoding consumes just $\sim 3J/s_v$. The difference is greater at 1080p, with software decoding consuming $\sim 25J/s_v$ compared

(a) ‘Big Buck Bunny’

(b) ‘Tears of Steel’

Figure 4: QoE measurements for both video sources at various picture sizes. The lines joining the points are visual aids only and do not imply interpolation to intermediate picture sizes.

to $\sim 10J/s_v$ using hardware assist. System resource usage (i.e. CPU and RAM) is also much reduced using hardware assist, leaving the system available to perform other tasks. However, the video card adds $\sim 25W$ to the system’s idle power. So, it is only when there is an energy saving of at least $25J/s_v$ that we gain from hardware assist.

Figure 5: Energy for video decoding of H.264 with and without hardware assistance. 20 runs were used to plot each point, with 95% confidence (error bars not always visible as they are very small in some cases). The lines joining the points are visual aids only and do not imply interpolation to intermediate picture sizes.

In future systems, energy savings could be gained by turning on hardware assist only when it would yield an energy benefit. This would require changes to hardware and software (e.g. drivers), and would also require new interfaces for the user to control this behaviour. A system profiling tool would need to execute in order to determine when the hardware control should be applied, the result being stored as part of a configuration file on the system, which is updated when system configuration changes (hardware or software).

6.3 Impact of energy savings

From Figures 2 and 6, we see how the differences in energy consumption of the codecs and picture sizes can be significant. For the same picture size, by using different codecs, we see that there are small and large differences in energy usage, e.g. $1J/s_v$ between FLV and H.264 at 360p for

‘Big Buck Bunny’, and $40J/s_v$ between FLV and H.265 at 1080p for ‘Tears of Steel’. Also, even with the same codec, we can see large differences with energy usage depending on picture size, e.g. $40J/s_v$ difference between 1080p and 480p for ‘Tears of Steel’ with H.265. This allows us to gain some measure of upper and lower bounds of savings that could be made by users exercising choice in the video stream they select. Our evaluation of the related impact is in the spirit of a Fermi estimate.

It is estimated that there are 6 billion hours of video streamed from YouTube every month⁶, a total of 72 billion hours per year, and we use this with UK energy consumption data in Table 7 to give an estimate of impact.

Table 7: Estimate of potential annual impact of energy savings based on 72 billion YouTube hours per year and UK energy consumption data.

J/s_v	KWh	Cost ^a	KgCO ₂ ^b
1	72M	GB£10.8M (US\$17.8M)	12.2M
40	2880M	GB£432.0M (US\$712.8M)	189.6M

^aFrom [29], mean cost of GB£577 / year per household.

^bUK government data, 10 May 2014, 0.17 KgCO₂/KWh⁷.

At energy savings of just $1J/s_v$ (the difference between FLV and H264 software decoding at 360p) for example, and assuming similar savings could be made on all client systems (in reality, it would vary), this would be enough to power 18947 homes in the UK for a year (mean annual home usage is 3800KWh [29]). In monetary terms, this represents $\sim \text{£}10.8\text{M}$ ($\sim \text{US\$}17.8\text{M}$) per year, and in terms of carbon emissions, 12.2M KgCO₂ emissions. The second row in Table 7 shows equivalent numbers for a reduction of $40J/s_v$ (the difference between using 480p instead of 1080p with H.265). Real savings will depend on the choices users make and the specific software, hardware and video sources, and will be subject to regional variations (e.g. the US emits 0.69KgCO₂ / KWh⁸, over 3 times as much as the UK). It is clear that the potential benefits economically and environmentally are significant at a global scale.

However, we believe our numbers in Table 7 are likely to be an underestimate by at least an order of magnitude because (i) we do not include other substantial video sources similar to YouTube (e.g. TED and news agencies such as the BBC), or movie services (e.g. Netflix, Amazon, etc.), or

⁶<http://www.youtube.com/press/statistics.html>

⁷<http://www.rensmart.com/Information/KWHToCO2Conversion>

⁸<http://www.epa.gov/cleanenergy/energy-resources/refs.html>

Figure 6: Derived energy metrics for decoding. 20 runs were used to plot each point, with 95% confidence (error bars not always visible as they are very small in some cases). The lines joining the points are visual aids only and do not imply interpolation to intermediate picture sizes.

real-time video (e.g. Skype, Facetime, etc.); and (ii) we do not assess the impact of transmitting fewer video bits across network infrastructure to the client and possible savings in network systems and devices [30]. There are also likely to be secondary impact factors, e.g. the energy use of content caches and proxies. Both of these will be significant when considered at a global scale. Also, different platforms will behave differently, e.g. on smartphones and tablets with hardware support for H.264, the situation is likely to be different, with other codecs decoded in software having much higher energy-consumption than H.264.

7. EMPOWERING USERS

Users must be given enough information to be empowered to make choices about video streams. We make two suggestions in this section, each with different levels of detail for client systems. Each scheme would allow users to make assessments based only on codec and picture size. So, for example, if a video stream provider only gives a choice of picture size and not of video codec, the user is still empowered with suitable knowledge to make a choice with respect to energy usage. As our methodology is measurement-based, we discuss first the use of a benchmark tool to provide data on the client system. Providing appropriate information in the correct format would require further analyses, additional experimentation, software development, testing and human factors work for real use.

7.1 System benchmark tool

Users are familiar with the use of benchmarks in order to assess the performance of equipment. For example, the Standard Performance Evaluation Corporation (SPEC)⁹ is a non-profit organisation that produces many benchmarks¹⁰

that are widely used for IT equipment and components, e.g. SPEC CPU2006 for CPU, SPECvirt_sc2013 for virtualisation, etc. SPEC also produces SPECpower_ssj2008¹¹, which assesses power usage using a suite of Java code that can be downloaded and executed on systems.

Consider a similar benchmark tool for video, which we shall call *vEQ-benchmark*. This would be downloaded by client systems, executed, and result in the creation of a system file containing information about energy usage of the system when decoding (and encoding) video. That is, it would be a suite that would produce a machine-readable summary, which we shall call *vEQ-summary*, based on the methodology presented in this paper, along with other data (e.g. video hardware information). The *vEQ-summary* file would contain information on codec types, estimates on energy usage for those codec types, QoE information, and other relevant information for users. It would be executed on system initialisation and when the system configuration changes. Especially at the larger picture sizes, there is a difference in energy usage between the two different types of content we have used in our experiments, both for encoding and decoding. So a benchmark would need to allow for such variation, e.g. by using a suite of reference video files.

SPECpower_ssj2008 relies on power analyser hardware to provide power information. However, future hardware would have power usage information available through an API much like the Advanced Configuration & Power Interface (ACPI)¹² already provides today for a range of systems.

Based on the contents of the *vEQ-summary* file, a video codec, video component (such as a browser plugin) could, along with information about the video, provide an interface to the user, to inform them about the energy usage and other information for a specific video file. A future applica-

⁹<http://www.spec.org/>

¹⁰<http://www.spec.org/benchmarks.html>

¹¹http://www.spec.org/power_ssj2008/

¹²<http://acpi.info>

tion adaptation mechanism could use that information along with user preferences to automatically adapt video quality to trade-off energy usage and QoE as required.

7.2 Simple user information

A simple way of presenting feedback to the user would be to indicate how ‘green’ a video file is, with a simple visual symbol, indicating the relative ‘greenness’ of a video file with respect to another. Figure 7 shows a mock-up for such an indicator from the YouTube video playback control bar in a web browser. The video plugin for the browser would, based on the contents of the vEQ- summary file, place green markers (a ‘leaf’ logo in this case) next to the different video formats that are already offered for playback. More green leaf markers indicate a ‘greener’ file.

Figure 7: A YouTube control panel (left), modified to show ‘green’ choices for video playback for the different picture size options (right). More green ‘leaf’ markers indicate a more energy efficient stream.

7.3 Detailed energy labelling

For many appliances, standard energy labelling is being used to indicate energy usage. For example, the “Directive 2010/30/EU of the European Parliament and of the Council of 19 May 2010”¹³ defines the indication by labelling and standard product information of the consumption of energy and other resources by energy-related products¹⁴. Many regions of the world have similar schemes and labelling. Such labels are now required on many appliances in the EU.

We propose that such a labelling scheme also be devised for video. However, rather than a printed label, a dynamic, adaptive label would be displayed on screen, using the vEQ-summary information to generate detailed energy and performance information. For example, in place of the YouTube display above in Figure 7, when a video file is selected the video file meta-data, along with the information from the vEQ-summary could be used to generate a label as shown in the mock-up of Figure 8. Qualitative indicators (top) as well as detailed quantitative information (bottom, for more technically aware users) allow users to make their selection. The qualitative indicators would be evaluated from using a combined quality/energy measure such as in Eqn. 7, using appropriate weights. Like other labels, e.g. for domestic appliances, our label contains other relevant information, such as QoE, file size, detailed video coding information, and availability of hardware assist.

¹³http://ec.europa.eu/energy/efficiency/eed/eed_en.htm

¹⁴<http://www.newenergylabel.com/>

Figure 8: A future interactive, video energy and performance label. A qualitative indicator allows quick and easy evaluation by the user. Detailed information at the bottom of the label could include: a QoE measure; energy usage; file size (downloading more bits uses more energy); and the availability of hardware assist for video encode and decode.

(Such a scheme could be applied to software components and systems more generally, not just video codecs, to complement labelling on hardware components and systems. This is an item for future study.)

8. CONCLUSION

Suitably informed users could provide economic and environmental benefits when using video. We have presented empirical results from our test-bed evaluations of the system energy usage of decoding and encoding of video streams. We examined seven popular codecs: FLV, H.264, H.265 (HEVC), MPEG4-II, Microsoft MPEG4-II (MSMPEG4), VP8 and VP9. Using a simple, measurement-based methodology, we have shown that the differences in energy usage between these codecs can be significant, e.g. a factor of 3 between decoding FLV and H.265 at a large picture size (1080p), and up to a factor of 10 for encoding. Both picture size and video codec choice impact energy usage. We presented simple energy metrics and demonstrated how these could be used with QoE to make an energy performance trade-off. We have also proposed two schemes to present energy usage information to users, to empower them to make choices on video usage. When considered across a global population of video users, individual client-side savings sum to show a potential for significant energy and cost savings, as well as environmental impact.

Acknowledgements

This work was partly supported by the IU-ATC project, funded by grant EP/J016756/1 from the Engineering and Physical Sciences Research Council (EPSRC). Oche Ejembi is funded by the Scottish Informatics and Computer Science Alliance (SICSA). Special thanks to Jim Park (School of

Computer Science, University of St Andrews) for modifying the Kill-A-Watt power meter.

9. REFERENCES

- [1] Cisco Systems, “Cisco visual networking index: Global mobile data traffic forecast update, 2012–2017,” Feb 2013. [Online]. Available: <http://goo.gl/OL6mYY>
- [2] W.-J. Hsin and J. Cigas, “Short Videos Improve Student Learning in Online Education,” *J. Comput. Sci. Coll.*, vol. 28, no. 5, pp. 253–259, May 2013.
- [3] M. Chen, S. González, H. Cao, Y. Zhang, and S. Vuong, “Enabling low bit-rate and reliable video surveillance over practical wireless sensor network,” *The Jnl of Supercomputing*, vol. 65, no. 1, pp. 287–300, 2013.
- [4] C. T. E. R. Hewage, M. G. Martini, and N. Khan, “3D Medical Video Transmission over 4G Networks,” in *Proc. 4th Intl. Symp. on Applied Sciences in Biomedical and Communication Technologies*, 2011, pp. 180:1–180:5.
- [5] T. Cai, E. Abbott, and N. Bwambale, “The Ability of Video Training to Reduce Agricultural Knowledge Gaps Between Men and Women in Rural Uganda,” in *6th Intl. Conf. on Information and Communications Technologies and Development: Notes - Vol. 2*, 2013, pp. 13–16.
- [6] M. Webb, “SMART 2020: Enabling the low carbon economy in the information age, a report by The Climate Group on behalf of the Global eSustainability Initiative (GeSI),” Global eSustainability Initiative (GeSI), Tech. Rep., 2008. [Online]. Available: <http://goo.gl/EOqSB4>
- [7] M. Sabharwal, A. Agrawal, and G. Metri, “Enabling Green IT through Energy-Aware Software,” *IT Professional*, vol. 15, no. 1, pp. 19–27, Jan 2013.
- [8] Y. Yu and S. N. Bhatti, “The Cost of Virtue: Reward As Well As Feedback Are Required to Reduce User ICT Power Consumption,” in *e-energy 2014 - 5th ACM Intl. Conf. on Future Energy Systems*, Jun 2014.
- [9] N. Amsel and B. Tomlinson, “Green Tracker: A Tool for Estimating the Energy Consumption of Software,” in *ACMCHI 2010 Extended Abstracts on Human Factors in Computing Systems*, 2010, pp. 3337–3342.
- [10] X. Li, M. Dong, Z. Ma, and F. C. Fernandes, “GreenTube: power optimization for mobile videostreaming via dynamic cache management,” in *Proc. ACM SIGMM 2012*, Oct 2012, pp. 279–288.
- [11] G. Nencioni, N. Sastry, J. Chandaria, and J. Crowcroft, “Understanding and Decreasing the Network Footprint of Catch-up Tv,” in *Proc. 22nd Intl. Conf. on World Wide Web*, 2013, pp. 965–976.
- [12] M. Song, Y. Lee, and J. Park, “CPU Power Management in Video Transcoding Servers,” in *Proc. NOSSDAV 2014*, Nov 2014, pp. 91:91–91:96.
- [13] Y. Sharab and N. Sarhan, “Detailed Comparative Analysis of VP8 and H.264,” in *IEEE Intl. Symp. Multimedia (ISM 2012)*, 2012, pp. 133–140.
- [14] D. Vatolin, D. Kulikov, and M. Arsaev. (2012, November) 8th MPEG-4 AVC/H.264 Video Codecs Comparison - Standard Version. [Online]. Available: http://www.compression.ru/video/codec.comparison/h264_2012/
- [15] A. Ravi and K. Rao, “Performance analysis and comparison of the Dirac video codec with H.264/MPEG-4 Part 10 AVC,” *Intl. Jnl. Wavelets, Multiresolution and Information Processing*, vol. 9, no. 04, pp. 635–654, 2011.
- [16] A. Rao, A. Legout, Y.-s. Lim, D. Towsley, C. Barakat, and W. Dabbous, “Network characteristics of video streaming traffic,” in *ACM CoNEXT 2011*, 2011, pp. 25:1–25:12.
- [17] Y. Liu, L. Guo, F. Li, and S. Chen, “An empirical evaluation of battery power consumption for streaming data transmission to mobile devices,” in *ACM SIGMM 2011*, 2011, pp. 473–482.
- [18] L. Zhang, F. Wang, and J. Liu, “Understand instant video clip sharing on mobile platforms: Twitter’s vine as a case study,” in *NOSSDAV 2014 - Network and Operating System Support on Digital Audio and Video Workshop*, 2013, pp. 85:85–85:90.
- [19] C.-H. Lin, J.-C. Liu, and C.-W. Liao, “Energy analysis of multimedia video decoding on mobile handheld devices,” *Computer Standards & Interfaces*, vol. 32, no. 1 - 2, pp. 10 – 17, 2010.
- [20] Y. O. Sharab and N. J. Sarhan, “Aggregate power consumption modeling of live video streaming systems,” in *MMSys 2013*, 2013, pp. 60–71.
- [21] R. Trestian, A.-N. Moldovan, O. Ormond, and G. Muntean, “Energy consumption analysis of video streaming to Android mobile devices,” in *IEEE NOMS 2012*, Apr 2012, pp. 444–452.
- [22] M. Tranchemontagne, “Moteino kill-a-watt: Hardware,” <http://goo.gl/TdoF5M>, 2013.
- [23] R. Serral-Gracià, E. Cerqueira, M. Curado, M. Yannuzzi, E. Monteiro, and X. Masip-Bruin, “An Overview of Quality of Experience Measurement Challenges for Video Applications in IP Networks,” in *WWIC2010 - 8th Intl. Conf. on Wired/Wireless Internet Communications*, 2010, pp. 252–263.
- [24] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE Trans. Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.
- [25] D. Vatolin, D. Kulikov, A. Parshin, and O. Petrov. (2006, January) MSU Subjective Comparison of Modern Video Codecs. [Online]. Available: goo.gl/3CpNWJ
- [26] L. Merritt, R. S. Bultje, and M. Niedermayer, “tiny_ssim.c, codec-independent objective quality measurement,” http://git.videolan.org/?p=ffmpeg.git;a=blob;f=tests/tiny_ssim.c, 2013.
- [27] S. N. Bhatti and G. Knight, “Enabling QoS adaptation decisions for Internet applications,” *Computer Networks*, vol. 31, no. 7, pp. 669–692, Apr 1999.
- [28] M. Tauber, S. N. Bhatti, and Y. Yu, “Towards Energy-Awareness In Managing Wireless LAN Applications,” in *NOMS2012 - IEEE Network Operations and Management Symp.*, Apr 2012.
- [29] “UK Energy Statistics, Statistical Press Release,” UK Dept. of Energy and Climate Change, Tech. Rep., Mar 2014. [Online]. Available: <http://goo.gl/gqxSWl>
- [30] M. Gupta and S. Singh, “Greening of the Internet,” in *ACM SIGCOMM 2003*, 2003, pp. 19–26.