Understanding and Decreasing the Network Footprint of Catch-up TV

Gianfranco Nencioni
University of Pisa
g.nencioni@iet.unipi.it

Nishanth Sastry
King's College London
nishanth.sastry@kcl.ac.uk

Jigna Chandaria
BBC R&D
jigna@rd.bbc.co.uk

Jon Crowcroft
University of Cambridge
Jon.Crowcroft@cl.cam.ac.uk

ABSTRACT

"Catch-up", or on-demand access of previously broadcast TV content over the public Internet, constitutes a significant fraction of peak time network traffic. This paper analyses consumption patterns of nearly 6 million users of a nationwide deployment of a catch-up TV service, to understand the network support required. We find that catch-up has certain natural scaling properties compared to traditional TV: The on-demand nature spreads load over time, and users have much higher completion rates for content streams than previously reported. Users exhibit strong preferences for serialised content, and for specific genres.

Exploiting this, we design a Speculative Content Offloading and Recording Engine (SCORE) that predictively records a personalised set of shows on user-local storage, and thereby offloads traffic that might result from subsequent catch-up access. Evaluations show that even with a modest storage of 32GB, an oracle with complete knowledge of user consumption can save up to 74% of the energy, and 97% of the peak bandwidth compared to the current IP streaming-based architecture. In the best case, optimising for energy consumption, SCORE can recover more than 60% of the traffic and energy savings achieved by the oracle. Optimising purely for traffic rather than energy can reduce bandwith by an additional 5%.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Network**]: Network Architecture and Design; C.4 [**Performance of Systems**]: [Measurement techniques]; H.4.3 [**Information Systems Applications**]: Communications Applications

Keywords

Catch-up TV; OTT content; workload characterization; energy savings; network footprint; traffic offloading.

1. INTRODUCTION

Recently, "Over-the-top" (OTT) Television services – distribution of Television (TV) content on the public Internet, without a dedicated network build out – have emerged as a new and cost-effective option. OTT distribution is especially popular for two kinds of content: streaming of movies (e.g. Netflix), and for providing so-called "catch-up TV" or "replay TV" service, which makes previously broadcast TV content available (typically via IP streaming) for *on-demand* or time-shifted viewing for a limited period after the original broadcast. Examples of catch-up services include Hulu in the USA and BBC iPlayer in the UK.

Unlike push-based traditional or "linear" TV, where programs are broadcast on different channels according to a known schedule, and user choice is limited to watching content showing on a channel at a given time, catch-up follows a pull-based approach, allowing users to choose what to watch and when to watch. Digital Video Recorders (DVRs, also known as Personal Video Recorders or PVRs) also support time-shifted viewing of linear TV by recording the broadcasts on user-local storage. However, users typically need to preprogram the DVR to record their favourite shows. In contrast, catch-up enables on-demand access and does not require users to anticipate what shows they want to have available for viewing later on.

Because of this flexibility, catch-up is becoming increasingly popular as an alternate or supplement to traditional TV in certain countries such as the UK. Ofcom, UK's independent regulator of communications industries, estimates that catch-up services are used by 44% of UK households [18]. According to Sandvine, BBC iPlayer is the most popular long-content streaming application in the UK, and second only to YouTube amongst all streaming video sources [19]. The UK market has several competing catch up TV services in addition to BBC iPlayer, thus the total impact of Catch-up TV on UK's networks is significant.

The traffic impact of Catch-up TV is likely to grow further as new devices such as smart and connected TVs become more prevalent. BBC recently observed over 7 million requests in one month alone from connected TV sets[1], representing an year-on-year increase of over 1000%. Today's accesses to BBC iPlayer are mainly from computer-based systems. Connected TVs, with their much larger screen sizes, will require higher fidelity video streams than required for PC and laptop screens, leading to higher average bit rates and thereby a larger traffic footprint on the network.

Additionally, the trend towards catch-up has increased the carbon footprint of TV content consumption. This is

WWW 2013, May 13–17, 2013, Rio de Janeiro, Brazil.
ACM 978-1-4503-2035-1/13/05.

[1] `http://www.bbc.co.uk/mediacentre/latestnews/2012/iplayer.html`

because broadcast has a fixed carbon cost which can be amortized across its viewers, whereas the carbon footprint of catch-up streaming grows with each additional user. The BBC estimates that for all of its channels except one[2], Digital Terrestrial Television (i.e., broadcast TV) has a smaller per-viewer carbon footprint than catch-up streaming [9].

Using historical access data from nearly 6 million users of the nationwide live deployment of BBC iPlayer in the United Kingdom, this paper seeks to understand how the nature of the catch-up TV workload affects its network footprint, by comparing with traditional linear TV and DVR usage. Our contributions are twofold. First, we show that the pull-based nature of catch-up leads to more manageable network footprint than linear TV, with better network utilisation and scalability. Second, we ask to what extent on-demand access can be converted to the pre-recorded time-shifted viewing model of DVRs by anticipating and predictively recording broadcasts likely to be accessed on catch-up, and show that this can be used to significantly decrease both the traffic and carbon footprint of catch-up TV.

We begin by characterising how people access content on catch-up, and what they access. The workload exhibits the following properties:

P1: Load spreading The support for time-shifted viewing is used extensively: Although content broadcast during TV prime time is also popular on catch-up, accesses are more distributed in time, resulting in a more even spread of peak time load.

P2: High engagement Users show a high engagement: the proportion of short-intervalled catch-up streams (i.e., streams abandoned or stopped after a short period of viewing) is relatively small. This is in contrast with the previously reported high-levels of short-intervalled viewing due to channel surfing in traditional TV.

P3: Strong preferences Users exhibit strong preferences. Shorter content of duration 30 or 60 minutes is more popular than longer duration content such as feature films. Serialised content items (TV shows broadcast as a sequence of episodes, typically one per week) are especially popular, as are certain specific genres.

These properties imply a natural scalability of catch-up that makes it easier to deliver over the public Internet than traditional linear TV. For instance, **P1** indicates that catch-up has better network utilisation than traditional TV, with less pronounced peaks and troughs in diurnal traffic patterns. Users typically finish what they start watching (**P2**), also leading to better network utilisation, with fewer "wasted" streams. Moreover, **P2** suggests that content delivery architectures could consider aggressively prefetching large chunks of content in advance of their view deadlines, when bandwidth and other network resources are available. **P3** can form the basis for choosing which subsets of content to cache, or sophisticated personalised content delivery architectures.

Next, taking advantage of the properties of high engagement and strong user preferences, we design and evaluate the Speculative Content Offloading and Recording Engine (SCORE), which attempts to convert on-demand catch-up accesses to DVR-like recording followed by time-shifted viewing. SCORE decreases the network footprint of catch-up by anticipating on-demand catch-up accesses, automatically recording broadcasts of such items on user-local storage such as DVRs, and then playing the local copy if accessed on-demand, thereby avoiding network usage.

Because the automatic recordings compete with user requested recordings for storage on DVR-like devices, we assume that only a limited storage is available to SCORE. As a baseline we use $S_0 = 32GB$, in line with reserved storage in standards such as YouView [23]. To carefully make use of the limited storage available, we take advantage of the strong affinity of users to certain types of content. SCORE consists of a predictor that assigns a personalised probability of viewing for each content item scheduled to be broadcast, and an optimiser that decides which items to store based on the probabilities and storage space available.

Our evaluations show that given access to just 32GB of storage, an oracle with complete knowledge of users' future accesses could, depending on parameter values of the energy model we use, the bit rate used for streaming, etc., save up to 97% of peak traffic, and up to 74% of the energy. For similar parameter values, SCORE is able to recover more than 60% of the energy and traffic savings obtained by the oracle. Dependency on parameter values is resolved using sensitivity analysis. SCORE is conservative in choosing items to offload because speculative recording of items not watched later increases energy consumption. If traffic is the only consideration, content can be offloaded more aggressively as there is no penalty for being wrong. This can achieve an additional 5% traffic reduction. The difference can be seen as the "price of being green".

The rest of the paper is organised as follows. §2 discusses related work. §3 introduces BBC iPlayer and clarifies TV terminology. §4 characterises properties of the BBC iPlayer workload and their implications for content delivery. §5 introduces SCORE and the details of its design. §6 evaluates the energy and traffic savings of SCORE in relation to the optimal savings achievable by an oracle. §7 concludes by discussing deployability concerns and limitations of our study.

2. RELATED WORK

This paper makes two contributions. The first is an characterisation of the catch-up workload. A number of seminal works [24, 13, 8, 12] have examined different forms of TV and video-on-demand delivery over the Internet. These range from walled garden IPTV architectures to P2P live streaming workloads. We add to this list by examining a catch-up TV workload. An important difference from previous work is that we find "scanning" or "channel zapping" behaviours to be much smaller than previously reported [24, 8], which we conjecture to be a result of the detailed Program Guide Information made available to users in our catch up system before they start streaming.

Our second contribution is the development of a simple algorithm to automatically store content that is likely to be watched by users later on. It has been recognised before that a large amount of savings can be realised by offloading content from the servers [14]. In walled-garden IPTV approaches, when the operator has control over the network, caching at appropriate locations and branch points within the network can be effective [21, 4, 6]. Deployments operating over the public Internet have to rely on end-users, and a popular strategy is to use P2P approaches where users collaboratively download from each other to de-

[2]The BBC Parliament channel, which has fewer viewers compared to other channels.

crease server load. However, supporting the delivery constraints of streaming in P2P architectures typically introduces complexity such as elaborate mesh/tree topology construction [16, 7], or careful chunk-scheduling strategies [3, 22, 10, 15]. Instead of peers, SCORE exploits the existing *broadcast channel* to decrease server and network load. While this makes the solution specific to catch-up TV, it also makes the design of SCORE very straightforward.

Functionality similar to SCORE is available on some commercially available DVRs, but there are differences. For example, some DVRs, such as TiVo, assist in content discovery by recommending *new* programs to watch [20]. Our goal is similar but with an important difference: we wish to learn the *existing* viewing habits of users and anticipate their usage of catch-up TV. TiVo essentially records as many relevant suggestions as possible, as low priority items to be erased if user-requested recordings require space. SCORE is much more conservative because recording content not watched later on wastes energy. Recent commercial offerings in the USA such as "Primetime Anytime"[3] from DISH, automatically record evening prime time shows for the four major broadcast networks during evening Prime Time. Sky TV in the UK follows a similar approach. The programs recorded by these offerings are expected to be the most popular shows. However, recording all the top-n shows for all users could yield negative energy savings (§6.2.1). For $n = 10$, which yields the best energy savings, the personalised SCORE approach outperforms the one-size-fits-all top-n approach both in energy and traffic savings.

3. PRELIMINARIES

This section introduces the BBC iPlayer, the data set we use, and TV-related terminology used.

3.1 TV content terminology

Television industries in different countries use slightly different terminology, and the same term sometimes has different meanings. Terms are used in this paper as follows:

TV shows, also known as *programs* (*brit.* programmes), are typically serialised into chunks known as *episodes*. Individual episodes, which we interchangeably refer to as *content items*, form the basic unit of broadcast. Usually, one new episode is broadcast per week, although the same episode can sometimes be broadcast at multiple times. A run of several connected episodes is known as a *series* or *season*; some programs can run for several seasons. Although there can be one-off broadcasts of content items (e.g., movies), we treat these as part of a program series with just one episode.

TV shows are assigned to a *channel* owned by the content provider or broadcaster. In traditional broadcast and cable TV, the channel represents the basic unit by which a user may select content for watching. At any time, a channel can only broadcast one content item; this introduces a linear schedule for content items. Thus traditional TV is sometimes known as *linear TV*.

Because the linear schedule is enforced on all users, an extremely important decision for traditional TV is deciding which content items to broadcast when on each channel. Audience numbers dramatically increase during a few hours in the evenings, known as *prime time*. Complex analytics including expected demographics, viewership numbers, their preferred times for viewing etc. may be used to decide which programs to fit into the prime time slots.

The broadcast schedule of a channel for the coming week is typically known in advance and is often available as an Electronic Program Guide (EPG). This information usually classifies the content item into well-known *genres*, such as drama, comedy, factual, and so on. Other information may include the names of actors involved, a blurb or summary describing the content item, etc.

3.2 TV content access in the UK

TV content can be delivered in many ways. To provide context for SCORE, which presupposes access to a DVR, we give a brief overview of common TV-related hardware and service delivery options in the UK: The first delivery mechanism is over-the-air broadcasts. This used to be analog signals, but has shifted to Digital Terrestrial Transmission (DTT) as with many other countries. A popular option to receive DTT broadcasts is using so-called Freeview or YouView boxes. These are sold without the need for a content subscription, and receive many over-the-air transmissions. All but the most basic Freeview model have an integrated hard drive-based Set-Top Box (STB); many can be connected to the Internet. Thus access to high-end DVR storage is available even to users without a cable connection.

Cable and satellite TVs are other common ways to receive content. Cable and satellite connections typically include a set-top box. Many STBs include a hard drive to store programs; this is sufficient for SCORE. Increasingly, an Internet connection is becoming a part of this setup, and is used to access either proprietary content on-demand from the cable/satellite provider or catch-up services from the content provider. Internet-connected TVs and TVs connected to games consoles may also include an iPlayer "app" and typically have local storage available.

3.3 BBC iPlayer

The BBC has a number of local and national TV and radio channels, which broadcast content over the air in the UK. The BBC iPlayer makes this broadcast content available on the iPlayer website for a fixed period of days after the broadcast, depending on content licensing terms and other policies. Thus, the iPlayer provides an alternate "over-the-top" access mechanism for content which is typically broadcast over the air. iPlayer additionally provides live streaming access to content currently being broadcast, but accessing a live video stream requires a TV license to be paid for annually. After the broadcast, the content is available for "catch up" viewing, and may be accessed even without a TV license, for viewers within the UK. Because of the free access after the broadcast, the BBC iPlayer is widely used within the UK as a catch up TV service; and to date has been used by an estimated 40% of online adults in the UK. Both live and catch-up iPlayer are entirely free of advertisements (ads) since the content programming is supported by TV licensing fees; user actions such as program abandonment may be attributed to content rather than ads and ad breaks.

3.4 Dataset

This paper studies a dataset derived from eight weeks of access logs to BBC iPlayer, from 04-Sep-2010 to 31-Oct-2010. Live streams have been filtered out, capturing on-demand or "catch up" access occuring after the broadcast.

[3] `http://dishuser.org/ptat.php`

Catch-up represents the majority of accesses. In September, for example, Live TV viewing represented about 10% of all accesses; catch up constituted the remaining 90%.

One in every four accesses to iPlayer during this period is recorded in the access log, giving a 25% sample of all accesses. This negatively affects our evaluation of the energy and traffic savings achievable by SCORE. We discuss this further in §7.1. However, all the measurements characterising the workload in §4 are percentages or fractions that are expected to be robust against a uniformly sampled workload.

The access logs are time stamps of the start and end of the streaming of one content item to one user. Altogether, the filtered trace consists of 32,691,343 streams from 5,985,458 users. These streams cover 37,728 unique content items (episodes) from 3,518 programs broadcast over 73 channels.

In addition, the BBC maintains web pages about each programme and episode which has been broadcast. By harvesting this data, we are able to augment the historical access logs with broadcast-related information such as the time and channel of broadcast, and the theoretical duration of the content item. We also identify each content item as belonging to one (or more) of eleven genres: kids, drama, learning, factual, music, news, religion and ethics (r&e), sport, weather, comedy and entertainment (entert.).

4. CHARACTERISING CATCH-UP

This section characterises the workload presented by catch-up access in the BBC iPlayer trace, and discusses the implications on designing content delivery architectures for catch-up. We first ask *how* users watch catch-up, and understand this in terms of what is known about linear TV. The main difference arises from the fact that content items may only be watched according to a pre-determined schedule in linear TV, whereas catch-up gives users control over when and what they watch. We then study in more detail *what* people watch, and use this as input to the design of SCORE.

4.1 How users watch catch-up TV

Choosing content to watch on catch-up TV is typically very different from traditional or linear TV. In linear TV, programs are broadcast at pre-determined times, and switching through channels (e.g. using a remote control) is thought to be the main method of choosing what to watch [8]. Although Electronic Program Guide (EPG) information (e.g. genre, duration, summary blurb, actors involved) is usually available, this is off the main content selection process. In contrast, catch-up is a form of video-on-demand; users are in direct control of *when to watch*, and choose *what they are going to watch* from a selection of available content. Further, detailed EPG information is usually available to users before they make their choice.

This section investigates the impact of giving users control over when and what to watch. We find that allowing users the choice of when to watch their content decreases the peak time load and spreads it out more evenly across time. We also find that program abandonment rates are smaller than previously reported for linear TV. We conjecture that this could be a result of users being in control of what they want to watch. Together, these two results imply that catch-up has better network utilisation characteristics than linear TV: fewer streams are wasted as a result of lower abandonment rates; and the network and content delivery servers have to be provisioned for a smaller peak because of load spreading.

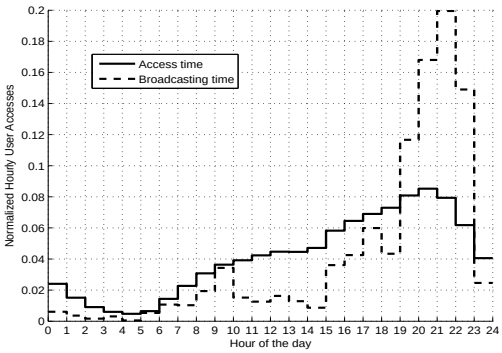


Figure 1: Normalised distributions of access times by hour of day, and the broadcast times of accessed items. Items broadcast during 7-11pm are very popular on catch-up, but the access hour is more evenly distributed.

4.1.1 On-demand access spreads load over time

First, we investigate *when* users access catch-up content, compared to broadcast times (Fig. 1). The dashed line in each time slot depicts the number of accesses (normalised by total number of accesses) to programs broadcast during the given hour of day. This clearly shows that items broadcast during evening hours (corresponding closely to TV "prime time") are much more popular than other content. This can be seen as confirmation that linear TV schedule has chosen the most popular shows for prime time, and that the same shows are popular both on linear and catch-up TV.

The dashed line also indicates the load distribution that would be seen if users were accessing content "live", as it is being broadcast. Instead, the solid line shows the actual (normalised) load distribution observed. This is much more evenly distributed, although a defined peak and trough is still observed, corresponding to users' evening free times and daytime work hours respectively. We conclude that on-demand spreads load over time, resulting in better network utilisation – a catch-up only service just needs to be provisioned for the lower peak, rather than the pronounced peak load that over-the-top live or linear TV needs to handle.

4.1.2 Program abandonment rates are low

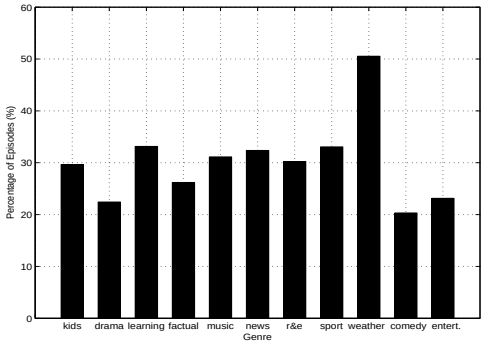


Figure 2: Percentage of items abandoned, by genre, showing relatively low abandonment rates for most, except weather.

In traditional TV setups, it has been shown that users mainly find items to watch by flipping through channels; this results in 60% of channel holding times being less than

10 seconds [8]. In contrast, users explicitly choose *what* to watch on catch-up, and can see a plot summary, actors involved, etc. before streaming. We wish to measure the effect of this on items actually streamed. We define a program stream as being abandoned if the users stops watching it in under five minutes. This threshold is somewhat arbitrary, but has been chosen to be low enough to accommodate most genres (e.g. many kids' programs are only 10 minutes long), and yet give users enough time for losing interest after starting to view an item, or realising that they have already viewed the episode before. On average, we find that ≈25% of catch-up streams are abandoned, much less than the >60% figure reported for linear TV. Fig. 2 shows the percentage of items abandoned, broken down by genre. Across the different categories, abandonment rates are in the region of 20–35%, with popular categories experiencing lower abandonment rates. The exception is weather-related content, which has a high abandonment rate of over 50%. It would seem that users only need the first few minutes of weather-related programs, or may be abandoning such programs for other reasons (e.g. they may skip to the part of the program relevant to their region, or only look for current weather rather than extended forecast).

Low program abandonment has implications for prefetching in different content delivery architectures: caching architectures can prefetch large chunks of content items likely to be watched. In SCORE below, we prefetch entire episodes, by recording them on local storage when they are broadcast. Similarly, progressive download-based streaming architectures, and peer-to-peer live streaming architectures may aggressively prefetch chunks which are due much later.

4.2 What users prefer to watch on catch-up

The previous section discussed the impact of giving users control over their watching pattern. This section asks *what* items they watch when given this control. We consider three axes of choice: duration of content, the type or genre of content, and whether the item is serialised, i.e., whether it belongs to a TV series comprising several episodes in sequence.

In each case, we use the same method to determine user preferences: we first consider the distribution of the parameter (e.g. content duration, genre or serial/non-serial) in the content corpus. Next, we consider a weighted distribution of the same parameter, weighted by the number of accesses. Their relative proportions for a given parameter value indicates user preferences: If a particular value of a parameter is overweighted in the weighted distribution compared to the content corpus, then users prefer that value. If underweighted, users dislike that value.

4.2.1 Users prefer serialised content

As a simple example of the above method, we find that serial content constitutes roughly 53.3% of the content corpus. Yet, in the list of items watched, serial content constitutes nearly 79.5%. Thus, we conclude that users prefer serial content items over non-serialised or one-off items.

4.2.2 Users prefer short duration content

Fig. 3 considers three distributions of content durations, *corpus*, *theoretical* and *actual*. *Corpus* is the distribution of content durations for each item in the catch-up content corpus. *Theoretical* is the distribution of durations obtained by weighting each item by the number of times it is accessed.

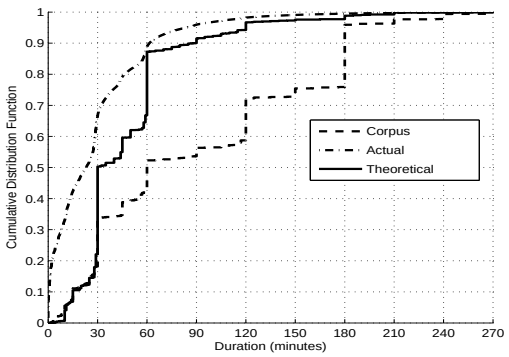


Figure 3: Content length distributions: *Corpus* shows the distribution of durations for all items in the content corpus. *Theoretical* is the distribution of content lengths for items watched. *Actual* shows the observed distribution of stream lengths. The content corpus has the most uniform distribution of content lengths. The theoretical distribution has nearly 90% of its mass under 60 minutes, showing that users prefer content shorter than an hour. Theoretical and actual distributions are close reconfirming low abandonment rates.

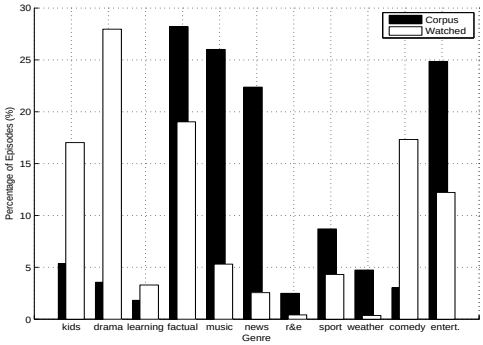


Figure 4: Distribution of episodes genres showing that drama, comedy and kids' programming are overweighted compared to corpus.

Corpus is much more uniformly distributed than *theoretical*, which has most of its mass under one hour. Observe further that the relative mass of *theoretical* increases dramatically at two points: 30 and 60 minutes, which corresponds to standard durations of serialised TV shows. This indicates the relative popularity of these two kinds of content. The third distribution, *actual*, gives the actual durations of streams observed. The difference between *theoretical* and *actual* is an indication of how much of the content is actually watched. This includes the ≈25% of requests abandoned in the first five minutes. Once this is excluded, the two distributions are even closer; suggesting high completion rates.

4.2.3 Users prefer specific genres

Next, in Fig. 4, we consider the relative proportions of different genres in the content corpus compared to their proportions when weighted by the number of accesses. Genres where the watched bar is taller than the corpus are overweighted, and hence preferred by users. This clearly indicates a strong preference for certain genres such as drama, comedy and kids' shows. In contrast, genres such as factual

programs, music and news constitute a large proportion of the content corpus but are not watched as much. Thus, although a public service broadcaster might provide a balanced content catalogue, users tend to prefer common kinds of entertainment over other genres.

4.2.4 Summary and Implications

In summary, we find that users overwhelmingly prefer serialised content, and short content of duration 30 or 60 minutes, and have a liking for certain specific genres such as comedy, drama and kids' programming. For content delivery architectures, these attributes can be used to decide which items to cache, if caching selectively. These user preferences can also form the basis for personalised content delivery architectures, as explored in the design of SCORE below (§5).

5. SCORE DESIGN

In this section, we present the design of our Speculative Content Offloading and Recording Engine (SCORE). First, in §5.1 we give an overview of how SCORE fits in, as an additional software element in DVRs. Next, we discuss the components of SCORE, the optimiser which decides the contents to store to minimise wasted energy (§5.2), and the predictor (§5.3), which prioritises content programs for the optimiser, based on expected affinity of the user to the program.

5.1 Overview of operation

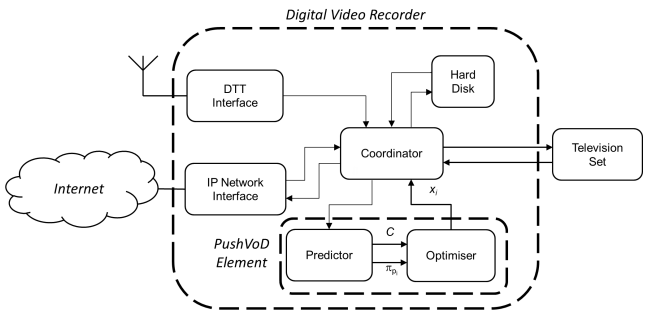


Figure 5: Schematic of a DVR/STB with SCORE

Fig. 5 shows a schematic of a DVR which includes the SCORE element. Content can be acquired either from the DTT interface during broadcast time, or pulled from the IP network interface. For each content item requested by a user, a coordinator decides whether to show the content from (a) the DTT interface (if the content is being broadcast live when the user requests to view), (b) the DVR (if the user had asked the content to be recorded, or if the SCORE element had speculatively stored the content) or (c) via IP streaming from the Catch-up TV servers, if not stored locally.

Apart from user-requested content items, the DVR also stores items speculatively when directed by the SCORE element. The SCORE element consists of a predictor and an optimiser. The predictor calculates weighting factors for each content item based on the program series to which it belongs. The decision on which items will be recorded speculatively is made by an optimiser, which calculates the expected utility of speculatively recording an item, subject to the storage limitations, and the other items that are due to be broadcast. The SCORE optimiser is run at the beginning of every week, using the upcoming broadcast schedule and the user's previous catch-up viewing history as inputs. The output is a schedule of content items to record speculatively. SCORE wakes up the DVR from sleep/stand by at the scheduled broadcast time, records the item, and goes back to sleep.

To avoid making anomalous predictions of items to record (which can waste energy if the recorded items are not watched later on), SCORE runs only after obtaining a sufficient history from the user. Currently, the threshold is set at 25 content items in the previous 3 weeks.

5.2 Optimiser

Speculative recording will never increase network traffic, but recording content not watched later on wastes energy. Although savings from watched items can compensate for unwatched items over a set of recordings, there can be net energy loss. Therefore we conservatively offload only content which is expected to minimise the energy spent in providing catch-up functionality (i.e., on-demand capability).

Deciding which items to record can be formulated as a *binary integer linear programming problem*. Formally, given a set of content items C that are known to be broadcast a given week, and a space constraint that a maximum of S bits can be stored, the task of the optimiser is to compute a binary valued variable $x_i \in \{0,1\}$ for each item $i \in C$. $x_i = 1$ if i is stored in the DVR, 0 otherwise. The decision is based on P^{IP}, the power consumption characteristics of the IP streaming option, P^{DVR}, the power consumed by the DVR for speculative recording, and the characteristics of the content item: the duration τ_i and the bitrate encoding r, which determine the space occupied, and a weighting factor $\pi_{p_i} \in [0,1]$ that encodes the probability that the user will watch item $i \in C$ based on the TV series p_i that i is part of.

Note that we model energy consumed in the Internet in terms of an energy *per bit* figure E^b, following Baliga *et al.* [5]. Although this model is based on a realistic paper design of a countrywide network, assuming data from commercially deployed networking equipment, there are, as with any such model, several approximations involved. §6.1 provides numerical details and discusses how we resolve the dependency on the E^b value by sensitivity analysis. In practice, for the storage levels we assume, the savings realised are relatively insensitive to E^b, especially for higher bit rates, indicative of future trends.

Speculative recording on the DVR can save energy only if

$$P^{IP} = E^b * r > P^{DVR} \tag{1}$$

However, speculative recording can still waste energy in either of two ways. First, the optimiser might decide to store an item which is subsequently never watched; thus wasting the energy involved in speculatively storing the item in the DVR. Second, the optimiser can decide not to store a content item which is subsequently streamed by the user.

The function of the optimiser is to minimise wasted energy expenditure while speculatively recording content. This is encoded in the following decision problem:

$$\text{minimize} \sum_{i \in C} \pi_{p_i} \cdot P^{IP} \cdot (1 - x_i) + \sum_{i \in C} (1 - \pi_{p_i}) \cdot P^{DVR} \cdot x_i \tag{2}$$

$$\text{subject to} \quad \sum_{i \in C} r \cdot \tau_i \cdot x_i \leq S \tag{3}$$

The objective function (2) is composed of two addends. The first computes the expected power spent for streaming items

which the optimiser decides not to store based on a probability of watching π_{p_i}. The second addend computes the expected power spent speculatively recording content which is not subsequently watched, based on the probability of not watching $1-\pi_{p_i}$. (3) imposes the constraint that the amount of stored contents must to be smaller or equal to the size of the memory S available on the DVR.

In theory, solving this decision problem accurately is NP complete. However, note that we can adopt greedy approach and select content items one by one in descending order of the objective function value (2) until we run out of space S. This works well in practice because most high probability content items are 30 or 60 minute programs; thus, this heuristic fills available storage except for a small slot usually < 60 minutes long.

5.3 Weighting factors

In (2), each program is weighted differently for each user, based on the probability of viewing the content. To be usable in the optimiser, the end requirement from a weighting model M is a weighting factor $0 \leq \pi_p^M(u) \leq 1$ for each user u and program p, with larger π_p^M indicating greater confidence that episodes of p will be watched via IP streaming. We develop different models which obey this convention.

The episodic nature of TV programs and the strong preference of users for serialised content gives a simple but powerful predictor: watching a large fraction of the previous episodes of a program is a good indication that the future episodes will also be watched. Formally, a weighting factor π_p^H can be derived for a user u who has previously watched n_p^u episodes of a program p with n_p episodes, as the frequentist probability of watching that program:

$$\pi_p^H(u) = \frac{n_p^u}{n_p} \tag{4}$$

A second possibility is to weight each program based on the affinity of the user to the genre(s) of the program. Adopting a vector space approach, we assign each user u a vector $\mathbf{g}_u = (g_u^1, g_u^2, \ldots, g_u^m)$, where g_u^j is the number of content items of the jth genre watched by the user. Similarly, each program p is assigned a vector $\mathbf{g}_p = (g_p^1, g_p^2, \ldots, g_p^m)$, where g_p^j is the number of episodes of p tagged with the jth genre. The genre-based weight π_p^G is then calculated as the cosine similarity between the user's genres and program's genres:

$$\pi_p^G(u) = \frac{\mathbf{g}_u \cdot \mathbf{g}_p}{\|\mathbf{g}_u\| \, \|\mathbf{g}_p\|} \tag{5}$$

We can combine this with the previous predictor as follows:

$$\pi_p^{G+H}(u) = \max(\pi_p^H(u), \pi_p^G(u)) \tag{6}$$

Clearly, more sophisticated predictors can be developed. In particular, we experimented with an item-item collaborative filtering-based predictor, that predicts new accesses based on similar items accessed in the past. However, we did not find any significant improvement over the simple predictors given above, and hence do not present its evaluation.

6. PERFORMANCE ANALYSIS

This section analyses the performance of SCORE using the trace discussed before (§3.4). We compute the aggregate energy and traffic savings achieved when SCORE is run by users in our trace, and present the results as percentage savings. We first discuss the simulation parameters used (§6.1). Then we assess the energy (§6.2) and traffic (§6.3) savings achieved for the network and the users by SCORE. In each case, we first use an oracle-based approach to compute the theoretical limits of the savings achievable by speculative recording. Next, the savings achieved by SCORE is measured relative to the oracle. Dependence on parameter values assumed is resolved by sensitivity analysis across the range of possible values for all parameter combinations. Finally, we evaluate how SCORE would perform if we optimise purely for traffic rather than energy savings (§6.4).

In computing the list of content items to speculatively record, we focus on weeks 4, 5 and 6 of our eight-week trace. This allows SCORE to work with the previous three weeks of history for the predictor, and at least two weeks after the broadcast for the user to watch the show, allowing a better estimation of achievable savings.

6.1 Parameters for trace-driven simulation

SCORE balances two factors which contribute to energy consumption other than on the content provider servers. The first factor is the energy consumed on DVRs to record the content. We conservatively consider HD double-tuner DVRs, which are the most energy-intensive of the simple Set-Top Boxes under EU regulations. [1] mandates a maximum power consumption of 13W when on or on active standby and to 1W when on passive stand-by for these devices. Therefore, the power consumption *added* by speculatively storing a content in the DVR, P^{DVR}, is conservatively taken as the maximum power difference possible between on and stand by states, i.e., 12W.

The second factor, the energy spent in the IP network to transport the content to the user, is much harder to quantify. Our use case of distributing content from a national broadcaster to audiences within the country over the public Internet closely fits the assumed model of Baliga *et al.* [5], which is based on a paper design of a national-level network in a broadband-enabled country, and includes a video distribution network for applications such as Video on Demand. The model makes detailed calculations using realistic numbers from various networking equipment currently deployed commercially, and provides a convenient method to calculate energy consumption parameterised in terms of E^b, the average energy per bit transported. However, as with other current energy models for the Internet, this introduces assumptions about the models and technology of networking equipment used, network hops from server to user, network over-provisioning and multiplexing levels, etc. To account for these uncertainties, Baliga *et al.* derive a range of values possible for this figure, from $E^b = 75\mu J$ for current networks down to $E^b = 2\mu J$, for a future energy-efficient all-optical network. Power consumed can be calculated as $P^{IP} = E^b r$ where r is the bit rate encoding of the content provider.

Given the inherent uncertainty and approximations involved in coming up with these values, we perform a sensitivity analysis over a range of values. The bit rate is varied as $r \in \{480, 800, 1500, 5000\} Kbps$. $r_0 = 800 Kbps$ represents the current default rate[4]; higher rates show currently available, and potential future encoding rates. We experiment with $E^b \in \{\frac{75}{8}, \frac{75}{4}, \frac{75}{2}, 75\}\mu J$, to see the effects over four (bi-

[4] `http://www.bbc.co.uk/blogs/bbcinternet/2009/04/bbc_iplayer_goes_hd_adds_highe.html`. However, when operating in full-screen mode on modern laptops, BBC iPlayer is seen to switch to 1500 Kbps.

nary) orders of magnitude. We do not go down to $E^b = 2\mu J$, the lowest value predicted by Baliga *et al.* [5], because at that point, for the bit rates we consider, $P^{IP} < P^{DVR}$, making it greener to stream than record on the DVR.

The amount of content that can be offloaded depends on the storage available on individual users' DVRs. Many current DVRs may have a 500GB or 1TB hard disk. Standardised technical specifications such as YouView DVR specify a minimum of 320 GB [23]. We assume that the SCORE element has access to a small fixed size partition in this space. As a baseline, we assume that a storage of $S_0 = 32GB$ is available, similar to the size of "reserved" partions in architectures such as YouView [23]. We refer to this as the *constant S* case. As the content encoding bitrate increases, fewer content items can be stored in a fixed size partition, leading to decreased gains. Therefore, we also experiment with a *rate-proportional S* case, where the partition size is taken as proportional to the bit rate encoding r as $S = S_0 \frac{r}{r_0}$.

6.2 Understanding energy savings

The energy benefits are quantified by computing the metric *Energy Savings* $= \frac{E^{IP} - E^{SCORE}}{E^{IP}} \cdot 100$, where E^{IP} is the energy consumption of streaming all the contents and E^{SCORE} is the energy consumption using SCORE.

We wish to understand energy savings at two levels. First, we quantify the potential of content offloading. Second, we measure the savings achieved by SCORE. However, we first examine the need for personalisation.

6.2.1 Understanding the need for personalisation

n:	1	10	20	50	100
% savings:	3.3%	4.6%	**-5.7%**	**-38.1%**	**-99.0%**

Table 1: Indiscriminately recording most popular n items for every user leads to negative energy savings ($E^b = 75\mu J$, $r = 800Kbps$, $S = 32GB$, week 6)

As a baseline, we first study a simple and straightforward approach to content offloading: offloading the most popular content to all users. Table 1 shows that doing so can lead to large numbers of unwatched items; recording items not watched wastes energy, resulting in decreased energy savings as n is increased. We see a net energy loss for $n = 20$ and beyond, motivating the need for a personalised, user-specific solution as developed by SCORE. The best energy savings are for a *top10* approach of saving the most popular 10 items for every user. §6.2.3 and §6.3.2 show that our personalised solution can indeed perform better than this baseline.

6.2.2 Oracle-based savings

To understand the full potential of content offloading, we consider the best-case scenario for a personalised solution: an oracle that has full knowledge of future content consumption decides offloads. Every item stored is guaranteed to be watched by the user. In this scenario, the achievable savings are limited only by the size of the storage available.

Fig. 6 shows the results, for different combinations of parameter settings[5]. The energy savings metric depends on E^b and r, which determine the power consumed by the IP streaming option, and S, which determines the amount of content that can be offloaded. Only those combinations where inequality (1) holds are considered; combinations of low r and E^b, known to result in negative energy savings, are not shown. In general, as E^b and r increase, IP streaming consumes more energy, and the energy savings are higher. However Fig. 6a shows that for very high bitrates, storage can become a limiting factor: The oracle is not able to store as many items as possible at lower bit rates, resulting in smaller energy savings (e.g. at $E^b = 75\mu J$, the savings from $r = 5000Kbps$ is smaller than savings from lower bit rates). Fig. 6b shows that this limitation is overcome when the storage is proportional to bit rate encoding. Fig. 6c shows the maximum savings achievable, by removing all storage constraints. If every item can be stored locally when broadcast, up to 97% savings can be achieved at high r and E^b. The maximum savings are $\approx 75\%$ considering a constant storage $S_0 = 32GB$, and $\approx 90\%$ considering a rate-proportional S.

[5]Error bars in all figures show 95% confidence intervals.

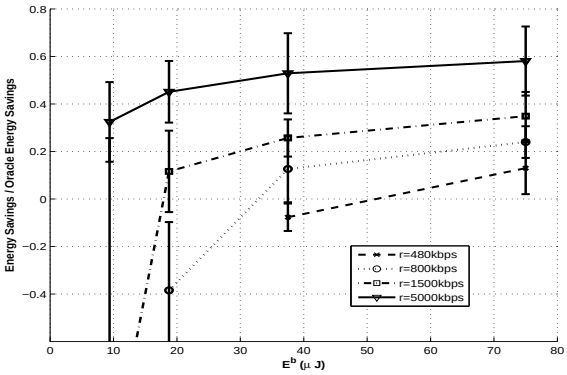


Figure 7: Energy savings of SCORE relative to oracle.

6.2.3 Energy savings in SCORE

Next, we study the savings achieved by SCORE, given access to $S_0 = 32GB$[6]. Fig. 7 performs a sensitivity analysis and shows the average energy savings by using SCORE for different combinations of parameter choices. For low values of r and E^b, the achievable energy savings are small, and errors in speculatively recording content which are not watched later can in fact lead to negative energy savings. However, at higher bit rates, savings appear to be relatively insensitive to the assumed values of E^b and SCORE can recover 40-60% of the optimal savings achieved by the oracle.

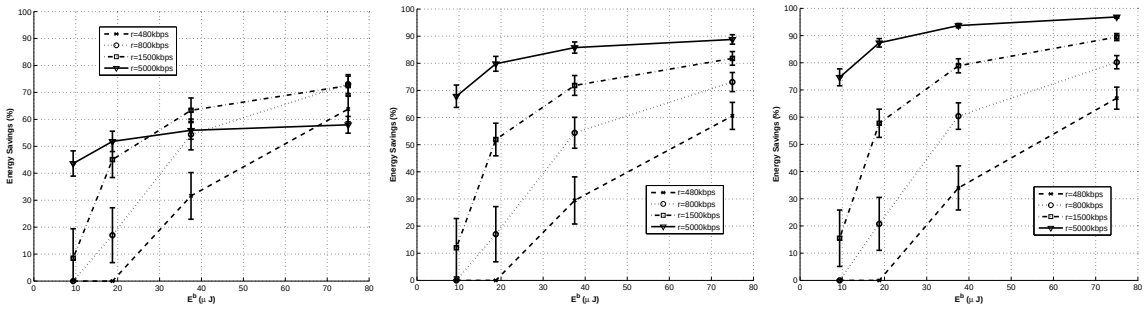
6.3 Understanding traffic savings

Next we study traffic savings by computing the metric: *Peak bandwidth savings* $= \frac{Q_{95}^{IP} - Q_{95}^{SCORE}}{Q_{95}^{IP}}$, where Q_{95}^{SCORE} and Q_{95}^{IP} are the 95^{th} percentile bandwidth taken across 5 minutes intervals by using SCORE and by streaming all the contents, respectively. This metric is intended to approximate the reductions in operating costs for ISPs, and uses the 95^{th} percentile bandwidth because many ISPs' Service Level Agreements (SLAs) are based on this figure.

6.3.1 Oracle-based savings

Fig. 8 shows the traffic savings obtained using an oracle with complete knowledge of future accesses. Unlike the energy savings computation, the oracle-based traffic savings do not depend on E^b, but only on r, the bit rate encoding,

[6]Due to space constraints, only the more challenging constant S case is presented for SCORE energy & traffic savings.



(a) Constant S (b) Rate-proportional S (c) With no storage constraints

Figure 6: Average energy savings (%) with oracle for different E^b, r and S parameter combinations.

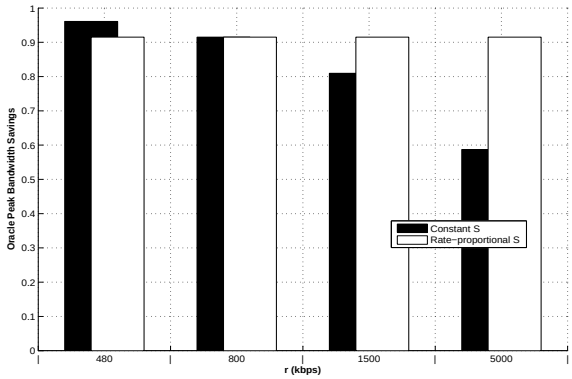


Figure 8: Peak bandwidth savings of oracle.

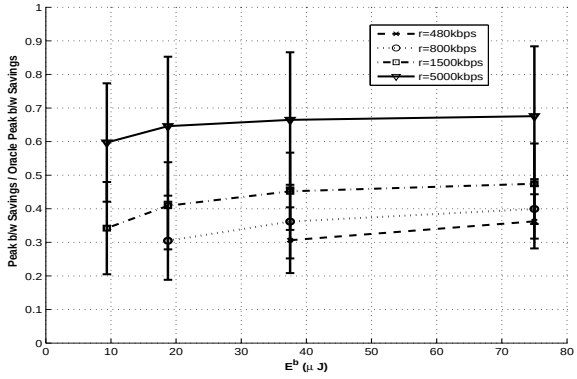


Figure 9: SCORE peak bandwidth savings relative to oracle

which determines the size of the IP flow, and S, the storage available on the DVR, which determines the amount of content which can be offloaded; an oracle with infinite storage can offload *all* the traffic. Therefore we only study the variation in savings for different values of r and finite values of S. The figure highlights that peak bandwidth is insensitive to the bit rate for rate-proportional S, because the memory size per content item remains constant across bit rates. Fig. 8 shows that the peak bandwidth savings can be up to 96% (i.e., peak bandwidth with the oracle can be as low as 4% of the peak without oracle-based offloading), but the peak bandwidth savings rapidly decreases when storage becomes a constraint (constant S scenario, for higher bandwidths).

6.3.2 *Traffic benefits from SCORE*

Fig. 9 shows a sensitivity analysis of the peak bandwidth

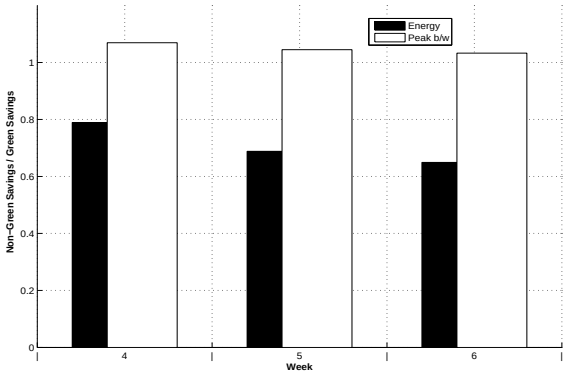


Figure 10: Comparing green or energy-conscious variant of SCORE (Eq. (2)) with the traffic-conscious or non-green version (Eq. (7)) shows that being green achieves 40% more savings in energy at the expense of only 5% more traffic. ($E^b = 75\mu J$, $r = 1500Kbps$, $S = 32GB$)

savings obtained by SCORE for different parameter settings. Note that unlike the oracle case, the savings with SCORE depend on E^b as well as r and S. This is because the items to download are decided as a *side effect* of saving energy (Eq. (2), also see discussion in §6.4). As with energy, SCORE typically recovers $\approx$ 40–60% of the traffic savings achieved by the oracle, using 32GB storage[6]. These savings are relatively insensitive to assumed values of E^b.

6.4 The price of being green

Eq. (2) decides which items to store speculatively based on their expected *energy* savings. Thus, even if storage is available, our implementation might decide not to save a content item because the expected energy savings may be negative, if our belief that the item will be watched (Eq. (6)) is low enough. If the content item ends up being watched, it represents a missed opportunity to save traffic. We evaluate this "price of being green", by changing the optimiser to the following "non-green" problem, which purely minimises the probability that a recorded content is not watched:

$$\text{minimize} \sum_{i \in C} \pi_{p_i} \cdot (1 - x_i) \tag{7}$$

subject to the memory constraint, Eq. (3).

Fig. 10 shows the impact of greening on the energy and traffic savings in terms of the ratio of the savings achieved in the energy aware or "green" case considered previously (Eq. (2)) to the savings achieved using the "non-green" case (Eq. (7)). The black bars show that the green solution

saves up to 40% more energy compared to the non-green solution.The white bars highlight that using energy-unaware SCORE, we could only achieve a traffic savings that is about 1.05 times greater, for the parameter settings indicated. This gap would be bigger if we consider lower values of E^b.

7. DISCUSSION AND CONCLUSIONS

We are currently witnessing the long-predicted convergence of IP and media networks in various forms. While this has offered additional functionality such as catch-up TV—the ability to watch TV programs on-demand, over the public Internet, without a dedicated infrastructure setup—the encroaching of TV content on the IP network can lead to additional network traffic and energy consumption.

Our contributions are twofold. Our first contribution is a characterisation of the catch-up TV workload, showing that catch-up has excellent network utilisation compared to traditional TV because on-demand spreads load over time and because stream abandonment rates are lower. Further, we showed that users prefer content shorter than one hour, and have a strong preference for episodic or serialised content, and certain genres such as comedy, drama and kids' shows.

Our second contribution is a simple approach that leverages the broadcast nature of TV, and the strong user preference for episodic and genre-based content to reduce the energy and traffic footprint of catch-up. The core of our proposal is to speculatively record content as it is being broadcast, using storage local to the user, such as those found on DVRs. Later requests for catch-up viewing can be served locally instead of incurring a network footprint. As a realisation of this concept, we presented the Speculative Content Offloading and Recording Engine (SCORE). For a given user, SCORE selects program episodes to record based on predicted affinity of the user to the program. We suggested simple predictors based on the affinity of users to serialised content and to specific genres.

Our main motivation in developing SCORE was to demonstrate that it is relatively easy to offload catch-up video streams from the Internet, and that significant savings can be obtained as a result. Below, we conclude briefly discuss two aspects that may need to be addressed for wider applicability of SCORE.

7.1 Better predictors for SCORE

Clearly, the performance of SCORE depends on the quality of the predictors used. In this work, we proposed very simple predictors based on episode histories, and genre affinity. We experimented with a more sophisticated item-item collaborative filtering approach, but for entire range of parameter values we assumed, no significant improvement was observed. Hence we present only the simpler alternative.

Our analysis may have been hindered by a few factors: First, other systems such as Netflix and TiVo which recommend items to watch per user can rely on users' ratings for previous content items watched. This information is not available to us. We can only rely on information that a user watched a particular TV show, but we cannot infer *how much* she liked it. Second, we are trying to predict which items will be watched on *catch-up* TV. This is a harder problem than predicting which items will be watched: some episodes may be watched on "catch-up" because the vagaries of a user's schedule might have prevented her from watching the episode as it was broadcast. Since our data only consists of catch-up TV, we will not know that the user regularly watches the TV show. We emphasise that an actual deployment will not have this problem, and can in fact record a show simply based on previous history of watching other episodes. Third, our dataset is sampled. This negatively affects us in two ways – if a historic access is missing due to sampling, we may be unable to predict a future access. On the other hand, we may predict an access that does happen in reality, but gets sampled out of the trace; our trace-based evaluation will not recognise the savings.

Despite these difficulties, we can, depending on parameter settings, recover more than 60% of the traffic and energy savings achieved by an omniscient oracle with full knowledge of a user's future consumption. We conservatively choose to offload only content which can save energy. Optimising purely for traffic can yield an additional 5% traffic savings. Furthermore, especially at higher bit rates which are indicative of present and near future catch-up deployments, these savings appear to be relatively insensitive to the assumptions we make about E^b, the energy per bit transported, the main parameter of the simplified energy model used.

7.2 Deployability and hardware requirements

As described earlier (§5.1), SCORE can operate as a software addition to DVRs. For wide applicability, SCORE needs to be deployable on existing DVRs, and DVRs should be widely available/used. Many DVR specifications include over-the-air or Internet-based software update mechanisms [23, 2]; these can be used to roll out the functionality of SCORE to existing DVRs. As noted in §3.2, many options to obtain TV content include DVR-like storage. Indeed, DVRs have over 50% penetration in major markets such as the US and UK [17, 11]. Thus, we believe SCORE has the potential to be widely deployed. Moreover, SCORE can be deployed independently by users; benefits to the system increase with each additional user, allowing incremental deployability.

The core functionality of SCORE, *offloading* content, requires only a basic DVR with user-local storage and a TV tuner. However, in order to seamlessly switch to IP streaming when the content is not available on local storage, an internet-connected DVR is required. Users with more basic DVRs need to manually switch to an Internet connected device as fallback.

In the current setting, many users who do not own a TV or DVR use the on-demand interface (e.g. Hulu, BBC iPlayer) through their laptop/desktops as their primary means for accessing TV content. For such users, a DVR-based solution may be too cumbersome. In principle, our solution only requires some form of user-local storage and the capability to record broadcasts; thus, SCORE can be deployed for computer-based users using a simple TV tuner attached to a computer. SCORE can operate as a daemon that selectively records content as it is broadcast, onto the computer's hard drive. Alternately, if saving *peak-period* traffic is the only consideration, the concept of SCORE can be used to selectively prefetch content to the laptop/desktop during night time, or other periods when spare bandwidth is available.

Acknowledgements

Jon Crowcroft would like to acknowledge the Engineering and Physical Sciences Research Council, UK, for funding programme grant EP/H040536/1. Gianfranco Nencioni was partially supported by pump-priming funds granted to Nis-

hanth Sastry by King's College London and by the GATECOM project funded by MIUR (Italian Ministry of Education, University, and Research).

8. REFERENCES

[1] Commission Regulation No 107/2009 of 4 February 2009 implementing Directive 2005/32/EC of the European Parliament and of the Council with regard to ecodesign requirements for simple set-top boxes. *Official Journal of the European Union L36* (2009), 8–14.

[2] Nordig unified requirements for integrated receiver decoders for use in cable, satellite, terrestrial and ip-based networks. Ver. 2.2.1, 2010.

[3] Annapureddy, S., Guha, S., Gkantsidis, C., Gunawardena, D., and Rodriguez, P. Is high-quality VoD feasible using P2P swarming? In *Proc. Intl. Conf. on World Wide Web (WWW)* (2007).

[4] Applegate, D., Archer, A., Gopalakrishnan, V., Lee, S., and Ramakrishnan, K. Optimal content placement for a large-scale VoD system. In *Proceedings of the 6th International Conference on Networking EXperiments and Technologies (CoNEXT 2012)* (2010), ACM, p. 4.

[5] Baliga, J., Ayre, R., Hinton, K., Sorin, W. V., and Tucker, R. S. Energy consumption in optical IP networks. *Journal of Lightwave Technology 27*, 13 (Jul 2009), 2391–2403.

[6] Borst, S., Gupta, V., and Walid, A. Distributed caching algorithms for content distribution networks. In *Proc. IEEE INFOCOM* (2010).

[7] Castro, M., Druschel, P., Kermarrec, A., Nandi, A., Rowstron, A., and Singh, A. Splitstream: high-bandwidth multicast in cooperative environments. In *19th ACM Symposium on Operating Systems Principles (SOSP)* (2003).

[8] Cha, M., Rodriguez, P., Crowcroft, J., Moon, S., and Amatriain, X. Watching television over an IP network. In *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement (IMC)* (2008), ACM, pp. 71–84.

[9] Chandaria, J., Hunter, J., and Williams, A. A comparison of the carbon footprint of digital terrestrial television with video-on-demand. BBC Research Whitepaper 189, March 2011.

[10] Choe, Y., Schuff, D., Dyaberi, J., and Pai, V. Improving VoD server efficiency with bittorrent. In *Proceedings of the 15th international conference on Multimedia* (2007), ACM, pp. 117–126.

[11] Deloitte. Technology, media & telecom predictions, May 2012.

[12] Dobrian, F., Awan, A., Joseph, D., Ganjam, A., Zhan, J., Sekar, V., Stoica, I., and Zhang, H. Understanding the impact of video quality on user engagement. *SIGCOMM-Computer Communication Review 41*, 4 (2011), 362.

[13] Hei, X., Liang, C., Liang, J., Liu, Y., and Ross, K. A measurement study of a large-scale P2P IPTV system. *IEEE Transactions on Multimedia 9*, 8 (2007), 1672–1687.

[14] Huang, C., Li, J., and Ross, K. W. Can Internet video-on-demand be profitable? In *Proc. ACM SIGCOMM* (2007).

[15] Huang, Y., Fu, T. Z., Chiu, D.-M., Lui, J. C., and Huang, C. Challenges, design and analysis of a large-scale P2P-VoD system. SIGCOMM '08.

[16] Kostić, D., Rodriguez, A., Albrecht, J., and Vahdat, A. Bullet: high bandwidth data dissemination using an overlay mesh. In *SOSP* (2003).

[17] Neilsen. Report: Bigger TVs, DVR and Wi-Fi among Hot U.S. Home Technology Trends, 2010.

[18] Ofcom. Communications market report 2012. Available from `http://stakeholders.ofcom.org.uk/binaries/research/cmr/cmr12/CMR_UK_2012.pdf`, July 2012.

[19] Sandvine. Global internet phenomena report, 1h 2012, May 2012.

[20] TiVo. How to find great new shows with TiVo Suggestions. Available from `http://www.tivo.com/mytivo/howto/getthemostoutoftv/howto_use_suggestions.html`, Last accessed 25 Apr 2012.

[21] Verhoeyen, M., De Vleeschauwer, D., and Robinson, D. Content storage architectures for boosted IPTV service. *Bell Labs Tech. J. 13*, 3 (2008), 29–43.

[22] Yang, X., Gjoka, M., Chhabra, P., Markopoulou, A., and Rodriguez, P. Kangaroo: video seeking in P2P systems. In *Proc. Intl. Conf. on peer-to-peer Systems* (2009), USENIX Association.

[23] YouView TV Ltd. Youview core technical specification, 2011.

[24] Yu, H., Zheng, D., Zhao, B., and Zheng, W. Understanding user behavior in large-scale video-on-demand systems. *ACM SIGOPS Operating Systems Review 40*, 4 (2006), 333–344.