VIDEO TRANSCODING TIME PREDICTION FOR PROACTIVE LOAD BALANCING

Tewodors Deneke[1,2], *Habtegebreil Haile*[1], *Sébastien Lafond*[1], *Johan Lilius*[1]

[1]Åbo Akademi University, Finland
[2]Turku Centre for Computer Science, Finland

ABSTRACT

In this paper, we present a method for predicting the transcoding time of videos given an input video stream and its transcoding parameters. Video transcoding time is treated as a random variable and is statistically predicted from past observations. Our proposed method predicts the transcoding time as a function of several parameters of the input and output video streams, and does not require any detailed information about the codec used. We show the effectiveness of our method via comparing the resulting predictions with the actual transcoding times on unseen video streams. Simulation results show that our prediction method enables a significantly better load balancing of transcoding jobs than classical load balancing methods.

Index Terms— Transcoding, Prediction, Machine Learning, Load Balancing

1. INTRODUCTION

Video content is being produced, transported and consumed in more ways and devices than ever. Meanwhile a seamless interaction is required between video content producing, transporting and consuming devices. The difference in device resources, network bandwidth and video representation types results in the necessary requirements for a mechanism for video content adoption. One such mechanism is called video transcoding. Video transcoding is a process of converting one compressed video representation to another. Currently transcoding is being utilized for such purposes as: bit-rate reduction in order to meet network bandwidth availability, resolution reduction for display size adoption, temporal transcoding for frame rate reduction and error resilience transcoding for insuring high quality of service (QoS) [1, 2].

Transcoding is a computationally heavy process and several methods has been proposed in order to increase its efficiency [3, 4]. Among them many attempts have been made to decrease its computational complexity through reusing information like DCT coefficients and the motion vectors extracted from the original coded data instead of fully re-encoding the video content. On the other hand to realize multiple transcoding and speed up, studies has been done to integrate multiple processors to fully decode and re-encode incoming video [5]. And more recently, new large-scale cloud based elastic transcoding architectures and services are emerging [6, 7, 8].

Runtime scheduling of transcoding jobs in multicore and cloud environments is hard as their resource requirements may not be known before hand. Currently for video transcoding jobs one has to rely on worst-case values which lead to an over provisioning of resources to maintain satisfactory QoS. This is due to the fact that the resource requirement of a transcoding job is highly dependent on the video data to be converted and its conversion parameters. In order to allow such distributed and multicore systems overcome the problem of over provisioning a method for predicting the resource requirement of each job is required.

Today, computing systems vary significantly from one another and range from very small (e.g. cellphones, tablets, notebooks) to very large (servers, data centres, cloud). However, at the heart of each of these systems there are resource management components that decide how to schedule the execution of different tasks over time (i.e., ensuring high system utilization or efficient energy use [9, 10]) or allocate program resources such as memory, storage and networking (i.e., ensuring a long battery life or fair resource allocation). These management components typically must be able to predict how a given task will perform depending on its size and its other characteristics, so as to decide how best to plan for the future. For example, considering a simple scenario in a cloud transcoding service with a set of two types of transcoding requests, fast transcoding jobs in set A and slow transcoding jobs in set B, A scheduler is often faced with the decision of whether to run each set on different CPU resources, potentially taking longer to execute; or to interleave between the two sets and distribute the job fairly, potentially executing the tasks much faster. If the scheduler can predict accurately how long each job would take to execute on a given platform, it can make an optimal decision, returning results faster, possibly minimizing energy, waiting time and maximizing throughput. Figure 1 shows an example distribution of video transcoding times on a set of randomly selected YouTube videos with randomly selected but valid transcoding parameters. Notice the heavy-tailed distribution in transcoding time values which will have a large impact on the performance of the system if scheduled improperly [11].

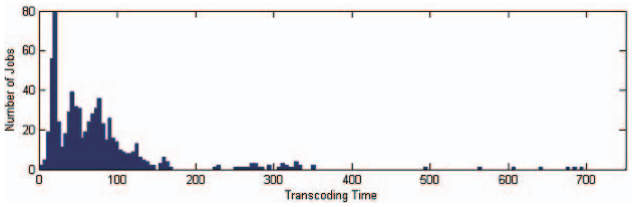

Fig. 1. Transcoding Time Distribution Over Randomly Selected YouTube Video Dataset

In order to leverage such opportunities, use of low overhead and accurate prediction mechanism is required. In this paper we present such prediction models trained based on an expert selected and easily obtainable video meta-data and transcoding (conversion) parameters. By basing scheduling decisions on such ahead of time knowledge, better resource utilization can be achieved.

2. RELATED WORK

Among others our work relates to video transcoding and prediction. In this section, we briefly summarize the related work on each topic.

2.1. Video Transcoding and Scheduling

Zhenhua Li et al. [12] implemented a cloud transcoder which requires a user to provide a video link and other transcoding parameters such as format, resolution, etc. Once a user provides the video link and other parameters, the required video is downloaded from the Internet and transcoded in a Cloud. After transcoding, the video is sent to the user. A copy of video is stored in a cloud cache to avoid repeated transcoding operations. The paper mainly focus on providing video transcoding service, and it does not talk about how such transcoding jobs are scheduled.

In [13] a distributed video transcoding was implemented with Message Passing Interface programming model. The video was segmented statically at Group of Pictures (GOP) level. The main focus of the paper was on parallelization and data distribution among computing units for bit rate reduction video transcoding. Although the paper provided a distributed video transcoding, however job scheduling was not the main topic.

In [8] prediction-based dynamic resource allocation algorithm to scale video transcoding service on a given Infrastructure as a Service cloud were discussed. The proposed algorithm provides mechanisms for allocation and deallocation of virtual machines based on a regression model that tracks and predicts the aggregate target transcoding rate required by the service. This work only uses queue length when load balancing transcoding jobs probably because tracking transcoding progress of individual streams is very expensive.

2.2. Machine Learning and Prediction

Roitzsch et al. [14] have presented per-frame decoding time prediction for modern video decoding algorithms. In this work they used expert selected metric to train and predict decoding time of videos encoded in MPEG. This work is specific to the MPEG family of codec and can not be applied directly to our problem as transcoding application should support a set of codecs from different codec families.

2.3. video characterization

There has been significant research on understanding the workloads of new generation video servers. These researches especially focus on the social aspect of videos and traffic characterization such as popularity, active life span, user access pattern, growth pattern, request patterns, etc.

Yu et al. [15] study user behaviour, content access pattern and their implications on the design of large-scale video-on-demand systems. Possible improvements on UGC design were proposed by Cha et al. [16] after studying YouTube and Daum, a popular UGC in Korea. After tracking YouTube transactions from a network edge, Gill et al. [17] have tried to understand video access characteristics and discuss the implications of their observation on key concepts such as caching. The caching problem in YouTube has been further studied by Zink et al. [18]. The social networking among videos was studied in the works of Halvey et al. [19] and Mislove et al. [20].

In this work we will reuse the traffic model from [15] to drive our experiments but further focus on collecting the missing statistics on video characteristics such as video length, size, bitrate, frame rate, codec type, resolution and etc that will be useful in our experiments.

3. SYSTEM OVERVIEW

Our work provides an approach for transcoding time prediction and shows its application in load balancing transcoding requests of a transcoding service. The main contribution of our work is to design an automated system that predicts the transcoding time of videos given an input video and a transcoding parameter set. These predictions can then be used for load balancing and QoS predictions by the service provider. Our system provides the opportunity for transcoding service providers to estimate the transcoding time of requests, and more intelligently manage their transcoding servers through proactive load balancing. We employ the idea of machine learning to design a framework that learns to predict transcoding time of videos. The key component of our work is to select fundamental video features that enable building precise transcoding time prediction model, and then use the resulting model on unseen videos to strategically load balance transcoding requests across multiple nodes. Figure 2 presents the overview of our framework. Typically transcoding service providers possess a log about transcoding requests (i.e. both transcoding parameter sets and the original video). Based on these traces, we can build a training dataset listing samples containing transcoding parameter set, the original video (or its fundamental characteristics such as resolution and bitrate) and measured transcoding time. Using such a dataset a prediction model can be trained via machine learning algorithms such as neural network and support vector machines (SVM). The model can then be used to predict and properly distribute load across transcoding nodes. The same prediction model can further be used to estimate the cost of transcoding a video and the QoS to be expected by the user.

4. TRANSCODING TIME PREDICTION

Machine learning techniques are often used as decision making mechanisms for a variety of systems. Basically, machine learning allows computers to evolve behaviours based on empirical data, in our case, this is a collection of samples with important video characteristics, transcoding parameter sets and measured

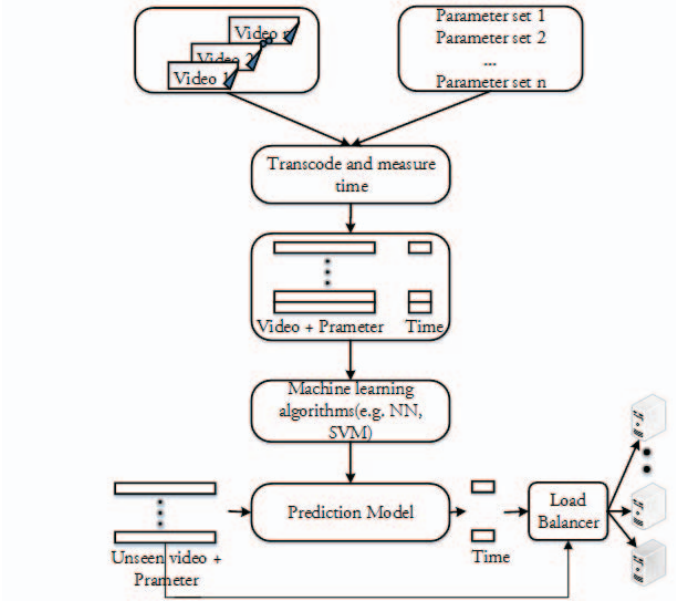

Fig. 2. Architecture of the prediction system

transcoding times. In this section, we present the detailed design of the proposed transcoding time prediction method based on the machine learning approaches.

4.1. Video Transcoding

An initial stage in building any prediction model is to understand the process itself, in our case video transcoding. The basic idea of video transcoding is to convert unsupported video formats in to supported ones. Unsupported videos include videos that are not playable by a given device due to luck of format support or those that require relatively higher system resources than the device can offer. The main types of video transcoding include, resolution transcoding, bitrate transcoding, temporal transcoding, container transcoding, codec transcoding , error reliance transcoding and any combination of these.

Resolution transcoding enables change of resolution of a given video allowing devices with lower or higher resolution to get served with the most appropriate resolution depending on the size of their display. Resolution of a video is defined by the number of pixels in its two dimensions. Transcoding rate and resolution have a clear correlation, the higher the resolution difference between the input and output video the more processing is needed and thus the lower the transcoding rate becomes.

Bitrate transcoding enables change of bitrate of the video allowing a given video to be served with the appropriate bitrate depending on the bandwidth capacity of the network the consumer device is connected to or storage media it has. Bitrate is one of the most important characteristics of a video stream. It indicates the number of bits in a video per unit time. Video transcoding rate correlates with the bitrate of the input and output video as it is directly proportional to the amount of bit that need to be processed. Larger bitrates enable higher quality but will require larger bandwidth and more processing power.

Temporal transcoding enables change of frame rate of a given video. The human visual system can perceives a sequence of more than 25 pictures per second as a video. This type of transcoding is sometimes used by video on demand providers to provide service with a lower frame rate (quality) in case of live events where encoding resources can become scares. Framerate indicates the number of frames (pictures) in a given video per unit time. The framerate of a video has a correlation with the transcoding rate of the video, the higher the framerate of the input or output video the lower the transcoding rate becomes.

Container transcoding also known as format transcoding is used to change the container (header) of a compressed video. Several container types such as flv and mp4 has been developed over the years to package video, audio and subtitle streams into one file. Each of these containers have some useful features that serve different purposes or are associated with a specific codec.

Codec transcoding Over the years different codecs (compression algorithms) have been developed. Usually the newer the codec the more advanced the algorithm it uses, allowing an ever increasing compression efficiency and quality. Example codecs that are being used today include h264, h263, vp8, and mpeg4. Backward (forward) compatibility with devices and infrastructures based on older (newer) codecs is maintained through codec transcoding. A video codec is a hardware or software implementation of video compression and decompression algorithms.

Video transcoding services provide parameters that control each of the transcoding types listed before. In addition to those parameters such applications (services) take the original video as an input which among others have fundamental characteristics such as bitrate, framerate, resolution, video duration, codec, frame types and count.

4.2. Dataset preparation and understanding

In order to build our model we need a training data. As we have discussed in the previous subsection transcoding time of a video is mainly dependent on a set of input and output video features. Based on expert knowledge we have partly presented in the previous subsection we have picked a set of features (metrics) that can be used in building our prediction model. This features include, *bitrate, framerate, resolution, codec, number of i frames, number of p frames, number of b frames, size of i frames, size of p frames and size of b frames* of the input video and the desired *bitrate, framerate, resolution and codec* of the output video which are given as a parameter to a transcoding service.

4.3. Modelling

To predict the transcoding time of a video, we use two of the most widely used supervised machine learning algorithms, the support vector regression (SVR) and Neural net. The fundamental idea behind any regression problem in machine learning al-

gorithms such as SVR and the Neural Net can be summarized as: given a set of t observations with n features (in our case the input and output bitrate, framerate, codec, etc) each and a target variable (transcoding time) y as $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_t, y_t)\}$ where $\mathbf{x} \in \Re^n$, $y \in \Re$ the objective is to find a function (model)

$$f(x) = \langle \omega, x \rangle + b = \mathbf{w} \cdot \mathbf{x} + b \text{ with } \omega \in \Re^n, b \in \Re \quad (1)$$

with the best fit as in equation 1.

Neural nets The idea of neural networks was first inspired by nervous system of human beings which consists of a number of simple processing units called neuron. Each neuron receives some input signals from outside or from other neurons and processes them with an activation function to produce its output and sends it to other neurons. These neurons can be understood as a mathematical function that take n element input vector and scale each data element x_i, by a weight w_j. The scaled data is offset by some bias b and put through a differentiable activation function such as equation 2. The output of a neuron can be analytically viewed as equation 1. The impact of Each input is weighted differently from other inputs thus a neuron is able to interpret the data differently depending on the weight and bias. Consequently the more is the weight the stronger the connection would be allowing that data point to influence the output more. The activation function f can be linear or non-linear. Non-linear activation functions are useful in mapping non-linear relationships. One such function is called sigmoid which is represented as

$$\frac{1}{1 + exp(-f)} \quad (2)$$

A network of these neurons forms a feed forward multilayer neural networks as in 3. These networks are made of layers of neurons. The first layer is the layer connected to the input data. After that there could be one or more middle layers called hidden layers. The last layer is the output layer which shows the results. One of the learning methods in multilayer perception

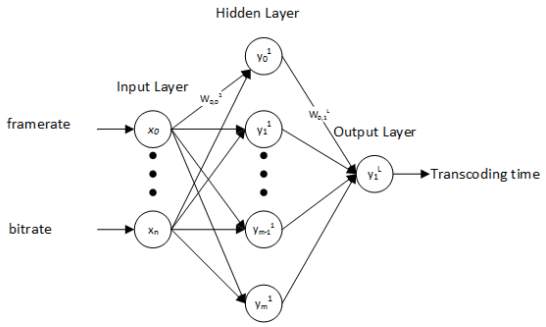

Fig. 3. Mluti-layer neural network used in our prediction

Neural Networks is the error back propagation in which the network learns the pattern in data set and justifies the weight of the connections in the reverse direction with respect to the gradient vector of error function which is usually regularized sum of squared error.

Support vector machines treats the regression problem as a convex optimization problem:

$$\text{minimize } \frac{1}{2}\|\omega\|^2 \quad (3)$$

$$\text{subject to } = \begin{cases} y_i - \|w, x_i\| - b \leq \epsilon \\ \|w, x_i\| + b - y_i \leq \epsilon \end{cases} \quad (4)$$

Similar to the neural net the SVR allows for non-linear solution through the use of radial basis function (RBF) kernels which are represented as

$$exp(-\frac{1}{2\sigma^2}\|f\|^2) \quad (5)$$

5. EVALUATION

5.1. Experimental Setup

To evaluate our proactive load balancing scheme, we simulate the throughput and job waiting time performance in CloudSim [21]. Every transcoding service provider has traces from its own log at its disposal. Although we do not possess such data, there are a number of sources (such as [15]) that provide information about web traffic in large-scale video-on-demand systems. In order to draw a realistic scenario, we imitate the video transcoding service request patterns with a shifted Poisson distribution as in [15]. We attempt to evaluate our proactive load balancing algorithm in a cloud environment. The largest provider of cloud infrastructure services in the world, Amazon, has published its virtual machine types, CPU types and the costs associated with them [22]. Under our CloudSim simulation environment, we deploy nodes, assign link capacities, and specify CPU and virtual machine characteristics according to data published by Amazon. The instance we used to base the characteristics of our CloudSim nodes is the c1.xlarge which is a 64-bit Intel Xeon E5-2680 machine with 8 virtual cores [22]. For generating the transcoding parameters and input source video associated with each request we used video characteristics data collected from YouTube.

5.1.1. YouTube Video Data Collection

For obtaining a realistic online video content statistics we use YouTube, the largest video sharing portal in the world. It is one of the most well known and widely used UGC (user generated content) website allowing users to upload, tag and share videos effortlessly. Users can also view, rate and comment on videos which brings a powerful social aspect to the site and in turn to its success. YouTube provides some statistics for its videos. Such information as view-count, number of likes/dislikes, duration and comments are public. However, more detailed statistics about fundamental video characteristics such as video bitrate, framerate, resolution and codec, which are essential to our experiment for generating realistic transcoding request parameters are not made publicly available by YouTube or any other similar service. Therefore we have implemented a crawler that uses the random prefix sampling method presented in [23] to sample over a million YouTube videos. For each video we sampled, we have probed, analysed and collected its fundamental characteristics using Ffprobe [24]. These collected video characteristics

are then used to generate valid transcoding parameters associated with our requests. For each transcoding request we simulate we obtained its actual transcoding time through measurement on a cloud instance (c1.xlarge) provided by Amazon EC2. Note that the characteristics of c1.xlarge is used to instantiate the nodes in our simulation environment.

5.2. Evaluating Prediction Accuracy

We evaluate the performance of our prediction accuracy by comparing it with measured transcoding time. First we have built a dataset containing input video, video transcoding parameters, and a measured transcoding time while transcoding the input video with the parameters on an Amazon EC2 c1.xlarge instance. The input video and the transcoding parameters are obtained through randomly picking videos from our crawled data from YouTube. Our dataset contains 2733 instances. We have divided this set into 2/3 training plus validation set and 1/3 test set. This means we have 820 instances of unseen test set. Finally we trained two prediction models based on neural network and SVR using the training and the validation set. In our Neural net model we have used 12 nodes in the hidden layer and the input layer takes 19 input attributes, 1 id and 1 label. The number of iteration for the back-propagation is set to 500. For our SVM model we used the grid search method to search for hyper-parameters using training and validation set. The number of support vectors used is 1795.

Figure 4 shows the prediction accuracy as compared to the actual transcoding time on the unseen test set. The result shows a mean absolute percentage error of 8.78% for our neural network model and a mean absolute percentage error mean absolute error of 18.64% for our SVR model.

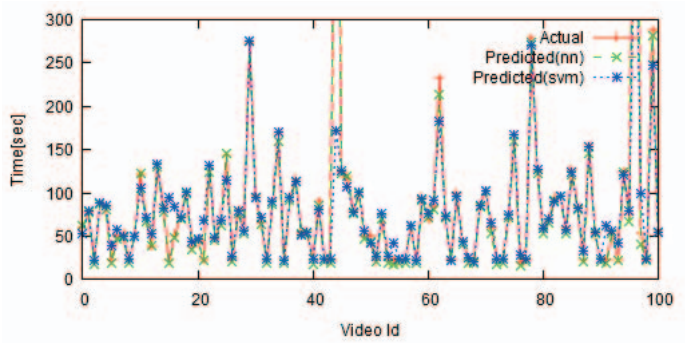

Fig. 4. Prediction Performance. Only 100 out of 820 results for figure readability

5.3. Evaluating Proactive Loadbalancing

We evaluate our proactive load balancing algorithms (see algorithm 1)in comparison with two widely used algorithms in video service systems. 1) queue length based load balancing. Load is being balanced based on queue length associated with each server. 2) Round robin approach, where transcoding requests are routed to transcoding servers in a round robin fashion.

Algorithm 1 Proactive load balancing algorithm

$server \leftarrow servers(0)$
for all req in $requests$ **do**
  for all s in $servers$ **do**
    if $predLoad(s) < predLoad(server)$ **then**
      $server \leftarrow s$
    end if
  end for
  $send(req, server)$
  $load(req) \leftarrow predictTranscodingTime(req, algo)$
  $predLoad(server) \leftarrow predLoad(server) + load(req)$
end for

Transcoding rate which measured in frames per second (fps) is defined as the number of frames transcoded per unit time (sec). Higher transcoding rate indicates better throughput, reduced delay and total cost of the system. Figure 5 illustrates the total transcoding rate of the system for a three hour load modelled from a realistic scenario described so far.

We observe that our proactive load balancing algorithms based on predicted load with our Neural Net and SVR models can achieve up to 15% improvement in terms of system throughput over the round robin and The queue length approaches before over provisioning occurs (i.e. 300 servers). One can also note that use of queue length or round robin produces the same result as the curves for these approaches overlap all the way.

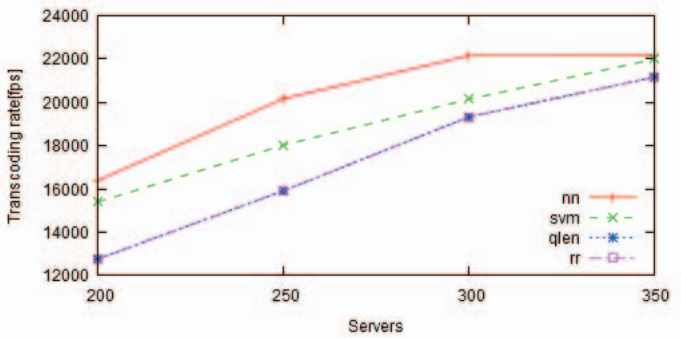

Fig. 5. Throughput

Average waiting time reflects the average waiting time of a transcoding request before it is given a resource and start being processed. Figure 6 shows the results from the simulation on the average waiting time of a job. Our proactive load balancing approaches enables reduction of average waiting time of a job by up to 18% enabling a better QoS.

6. CONCLUSION

This paper addresses the challenges exposed by the modern online video transcoding services. We design a proactive load balancing scheme to aid throughput and quality of service for video transcoding services. Our novel method explores the opportunities provided by trends from transcoding load of requests. Due

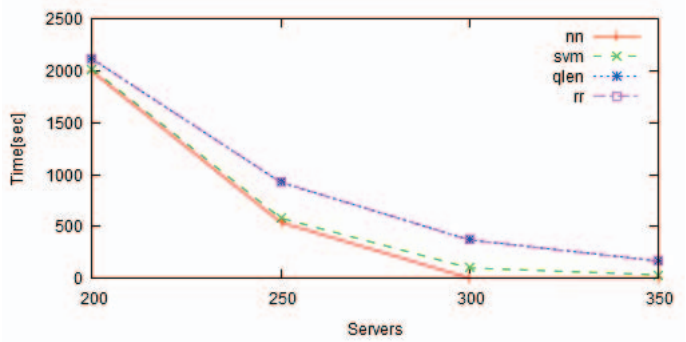

Fig. 6. Average waiting time for different load balancing approaches

to the correlation between transcoding time and transcoding parameters, we can predict transcoding time of a video given the fundamental characteristics of the input video and the transcoding parameters which in turn is used to properly load balance video transcoding requests across servers. In our experiment, we have used real-world data and designed the simulation scenario imitating real world Internet transactions. Our proactive load balancing algorithm shows effectiveness and significant improvement over the traditional methods. For the future work, we would like to further explore the effect of our load balancing on dynamic resource provisioning.

Acknowledgment

This work was supported by the Cloud Software Finland research project and by an Amazon Web Services research grant.

7. REFERENCES

[1] S. F. Chang and A. Vetro, "Video adaptation: Concepts, technologies, and open issues," *Proceedings of IEEE*, 2005.

[2] J. Xin, C.-W. Lin, and M.-T. Sun, "Digital video transcoding," *Proceedings of the IEEE*, 2005.

[3] G. Morrison, "Video transcoders with low delay," 1997.

[4] Jeongnam Y., Ming-Ting S., and Chia-Wen L., "Motion vector refinement for high-performance transcoding," *IEEE Transactions on Multimedia*, 1999.

[5] Y. Sambe, S. Watanabe, Dong Yu, Taichi N., and Naoki W., "High-speed distributed video transcoding for multiple rates and formats," *IEICE - Trans. Inf. Syst.*, 2005.

[6] Amazon Inc., "Amazon elastic transcoder," August 2013.

[7] Zencoder Inc., "Zencoder cloud transcoder," August 2013.

[8] F. Jokhio, A. Ashraf, S. Lafond, I. Porres, and J. Lilius, "Prediction-based dynamic resource allocation for video transcoding in cloud computing," in *PDP*, 2013.

[9] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, aug 2003.

[10] M. Isard, V. Prabhakaran, J. Currey, U. Wieder, K. Talwar, and A. Goldberg, "Quincy: fair scheduling for distributed computing clusters," SOSP 2009.

[11] Mor Harchol-balter, "The effect of heavy-tailed job size distributions on computer system design," in *In Proc. of ASA-IMS Conf. on Applications of Heavy Tailed Distributions in Economics*, 1999.

[12] Z. Li, Y. Huang, G. Liu, F. Wang, Z. Zhang, and Y. Dai, "Cloud transcoder: Bridging the format and resolution gap between internet videos and mobile devices," in *22nd ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2012.

[13] F. Jokhio, T. Deneke, S. Lafond, and J. Lilius, "Bit rate reduction video transcoding with distributed computing," in *PDP*, 2012.

[14] M. Roitzsch and M. Pohlack, "Principles for the prediction of video decoding times applied to mpeg-1/2 and mpeg-4 part 2 video," in *RTSS*, 2006.

[15] H. Yu, D. Zheng, B. Zhao, and W. Zheng, "Understanding user behavior in large-scale video-on-demand systems," *SIGOPS Oper. Syst. Rev.*, 2006.

[16] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn, and S. Moon, "I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system," IMC 2007.

[17] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "Youtube traffic characterization: a view from the edge," IMC 2007.

[18] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Watch global, cache local: Youtube network traffic at a campus network - measurements and implications," Tech. Rep., 2008.

[19] M. Halvey and M. Keane, "Exploring social dynamics in online media sharing," WWW 2007.

[20] A. Mislove, M. Marcon, K. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," IMC 2007.

[21] R. Calheiros, R. Ranjan, A. Beloglazov, A. De Rose, and R. Buyya, "Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw. Pract. Exper.*, 2011.

[22] Amazon Inc., "Amazon ec2 instance types," August 2013.

[23] J. Zhou, Y. Li, V. Adhikari, and Z. Zhang, "Counting youtube videos via random prefix sampling," IMC 2011, ACM.

[24] Stefano Sabatini, "Ffprobe," August 2013.