Asian Journal of Information Technology 15 (14): 2473-2482, 2016
ISSN: 1682-3915
© Medwell Journals, 2016

A Novel Cloud Based Scheduling Strategy to Perform Transcoding for H.264 Real-Time Streaming

D. Preetha Evangeline and Anandhakumar Palanisamy
Department of Computer Technology, MIT Campus, Anna University, Chennai, Tamil Nadu, India

Abstract: Recent advancements in cloud computing serves as a solution for many real world problems and one best application is the streaming system. Customizing, accessing and sharing of the multimedia files which are procured in the cloud is not alone streaming. Technically there are many other issues that have to be concentrated while streaming live contents. High-definition video applications are often challenging for mobile devices due to their limited processing capability and bandwidth-constrained network connection. The proliferation of cloud based educational system has raised the requirements for efficient representation of video where previously standardized video coding technology can hardly keep pace. Especially in live streaming, transcoding needs to be carried out on the fly with acceptable speed to maintain better quality of service. The objective of this paper is to propose a cloud based transcoding system using MapReduce to perform parallel operations, balancing the load of VM's. The study concentrates on NP-hard problem which aggregates job assignment overhead and scheduling based on capacity of the machines. There are two contributions proposed in this study, first is an algorithm to rule out redundancies in similar frames. A highly efficient algorithm called Extended Mode Prediction for Transcoding (EMPT) is proposed which takes care of the transcoding part. The second is a novel low complexity heuristic algorithm called Optimal Reduced Finish Time (ORFT) to minimize the overall finish time taken for transcoding. By considering the average complete time as the lower bound, an optimal solution is framed for the above mentioned problem. Implementation has been carried out using mathematical simulations and analysis to prove that the proposed algorithm outperforms better than the existing algorithms with a low complexity of O (nlogn).

Key words: E-learning, map reduce, cloud computing, heterogeneity, transcoding, load balancing, scheduling

INTRODUCTION

In recent years, there is a growing popularity of live streaming applications such as tele medicine, video conferencing, E-learning and so on. Now a days students are interested in lectures, tutorials and group discussions online through their mobile devices as it is portable, instant and makes life easier. Existing works concentrates on video-on-demand where the lectures course contents are downloaded and viewed based on their interests. But when it comes to live lectures across various institutions, it is not possible for most of the participants to view it online without interruptions.

When compared to video-on-demand, live streaming is much more sensitive to quality of service. Delivering contents is not just enough, delivering it based on the specification of device within a time constraint along with high definition is vital. Scalability issues, heterogeneity, load balancing along with quality of service needs attention when it comes to real world applications (Cha *et al*., 2007). This is where the coding technology comes into picture. Video coding still exist as a challenge due to varying range of requirements from video characteristics such as spatiotemporal resolution, chroma format, sample accuracy and applications ranging from videoconferencing and E-learning over mobile devices Yang *et al*. (2010). Usually the codec varies with varying environment and however complex it may be, the transcoding speed remains to be an uncompromised issue. When this challenge is taken into account, immediately the idea which strikes is the parallel processing. Processing in parallel generally reduces the time in completion of tasks. In spite of multi-core processors that supports parallel transcoding, it is hard to extend the process due to hardware specifications. Here comes the cloud computing in act where it consist of bundle of computing resources such as heterogeneous machines organized in a distributed fashion and can indulge in processing complex tasks parallel to overcome hardware specification challenge (Evangeline and Anandhakumar, 2015).

Corresponding Author: D. Preetha Evangeline, Department of Computer Technology, MIT Campus, Anna University, Chennai, Tamil Nadu, India

MapReduce model in our distributed environment is used for splitting up our input video and analyze the complexity of the task to assign them to “n” number of machines with different computing power. The Mapping module handles these tasks in multiple computers and the Reducer module collects the output from provided by the Mapper and combines them to give out the output according to the device specification. Hence, scheduling of jobs to its exact machine reduces the Overall transcoding finish time and the output is obtained much faster.

Now a days, users are very much interested in watching videos over mobile devices which is much more compatible when compared to laptops and computers Liu *et al.* (2012). Mobile device due created a greater hike inculcating latest technologies which has pulled users interest. One challenge here is the specification of the devices which requires live transcoding to be carried out in order to provide quality of service while streaming videos.

Zhu *et al.* (2013) speaks about a transcoding method Simplified Bidirectional Intra-Prediction technique. This approach aggregates two modes or factors out of a total of nine in the H.264 AVC standard with a specifically computed weight matrix. The values of the weight matrix depends on the modes chosen and also the distance between them. They have also illustrated that the number of possible combinations of these modes comes to 36. They have aimed to achieve a reduction in the computational time by adopting this strategy. Sullivan *et al.* (2012) proposed an intra-coding technique that helps to preserve high quality of the video and also takes a very much lesser computational time. Liu *et al.* (2013a) proposed the edge-oriented mode for predicting strong edges through different directional intra prediction modes.

“Cloudstream” a proxy based cloud model for transcoding live stream is enhanced with two mapping procedures. One is to concentrate on how to optimize the speed of transcoding and the other procedure is to reduce the jitter delay. Here time to assign the task to the machine is not considered.

Analysis of internet streaming service to mobile device specifies transcoding in the server side where videos can be converted into 40 different versions and streamed to the devices Balsree *et al.* (2005). Liu *et al.* (2013b) concluded that there exists a great demand of CPU resources when transcoding takes place online. Better replacement policies could be adopted for enabling optimized mobile streaming system (Liu *et al.*, 2011).

Cloud assisted SVC was proposed to handle the heterogeneity of the network. Here cloud gathers the information about both video resource as well as status about the network. The proposed Scalable Video Coding system encodes the video streams and stores it in multiple servers (Huang *et al.*, 2011). The video server on receiving the request for streams forwards the query to the cloud. Here the cloud computed multi-path strategy and provides labels for individual streams. The limitation of this work is, every time the network status being advertised may lead to overhead (Zhu *et al.*, 2013).

Han *et al.* (2014) CMTS Cloud Multimedia Transcoding System which overcomes the limitations of Cloud Multimedia Data Conversion System (CMCS) such as performance delay and transcoding failure. Transcoding failure tends to occur when large volume of video files could not be converted. In the proposed CMTS, Hadoop Distributed File System and MapReduce technology is used. Here the video files are created through various heterogeneous device and they are transferred to CMTS through web interface. The files are divided and stored according to HDFS policies. Predefined format and size that are specified by the user are accounted and the video files are transcoded. The advantage of this system is parallel processing that takes place on the divided streams. The limitation of the work is there isn't any method discussed at the time of Churn failure.

Every time a MapReduce model is used to minimize the complexity of computing, but in this paper it has been used in a cloud based transcoding system which manages the input video segments of varying length and complexities. These video segments are considered to be the sub tasks that are to be assigned to machines thereby matching the complexity with computing power. As stated above, the problem is formulated as a NP-hard which considers the job assignment overhead and Re-scheduling strategy of the residual segments. The paper proposes a novel Optimal Reduced Finish Time (ORFT) algorithm which comprises of two procedures:

- Knapsack procedure
- Re-assignment procedure

To validate the proposed algorithm mathematical simulations and analysis of the algorithm is been carried out.

System flow: The overall system flow of the proposed cloud based MapReduce transcoding architecture is explained under this section. Figure 1 shows the overall proposed architecture. This study concentrates on live streaming, the input video enters in the form of chunks with varying size, length, content and complexity. With these video chunks pre-processing operations are

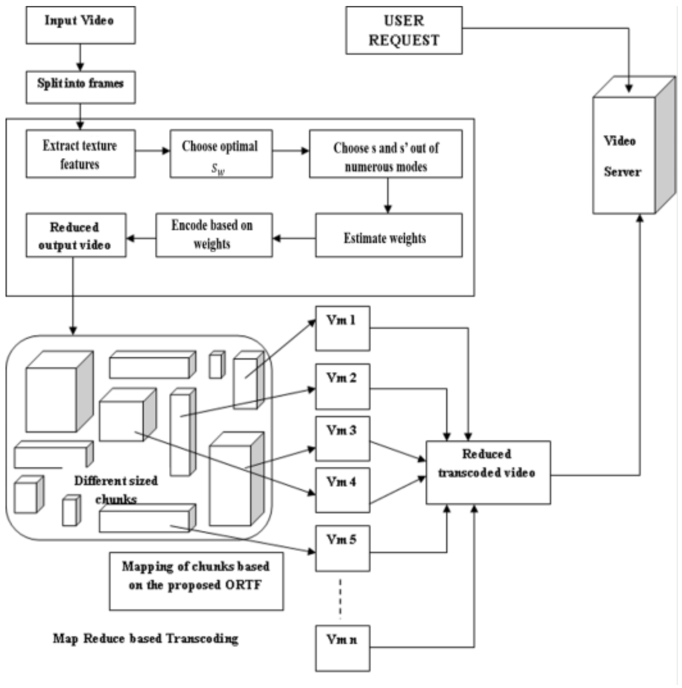


Fig. 1: Proposed cloud based transcoding architecture prediction

performed. Pre-processing consists of splitting up of chunks into frames, extract the key features based on texture, applying the proposed prediction based transcoding algorithm to reduce the size of the video chunk as well as maintaining the original quality of the video by smartly coding them.

After pre-processing is done, complexity of the video is estimated. Complexity is determined in terms of length, spatial details, luminance and chrominance components. The video segment is split further into "n" blocks with varying complexities and based on the MapReduce model it is passed through the scheduler which uses ORFT algorithm that maps the block "b" to the machine "m" with the computing power "p". The computing capacity of each individual machine is calculated based on its processor speed and its historical transcoding records. The subtasks are processed sequentially in every machine without preemption. The overall transcoding operation finishes once all the machines complete executing the task that is assigned to them. The consolidator rejoins the segment and the output is delivered to the user.

Problem statement: In our problem statement only the transcoding operations in Cloud environment is considered and Job assignment overhead is considered in the scheduling strategy. Assume there are "n" different blocks B = {b1, b2, b3,…..bn} with different complexities C={c1, c2, c3,…..cn}. Here each segment is executed without any preemption by other segment. We consider "m" computer with different power M= {p1, p2, p3, ……pm} and the job assignment overhead = $J_{Overhead}$. Here the Transcoding time depends on the complexity of the segment which proves the time taken for transcoding is directly proportional to the complexity. Transcoding Time spent for a block „b" on a computer with power p" can be computed as:

$$t_{hn} = C_b / M + J_{overhead} \tag{1}$$

Every computer would consist of several segments where Sp is the set of segments on computer "p". The completion time of Sp is:

$$C_{SP} = \sum\nolimits_{b\in SP} t_{bp} = \sum\nolimits_{b\in SP} C_b / M_p + |S_p| \times J_{overhead} \tag{2}$$

The scheduling strategy can be denoted as G = {S1, S2, S3,...Sn} and The assignment of block "b" is given as:

$$A_h = B \rightarrow G \tag{3}$$

Which means the assignment of the segment b is Ab and mapping of segments to machines is denoted as B→G. The objective is to find the scheduling strategy that minimizes the entire complete time, bounded by maximal complete time of all computers:

$$\min_G \max_{Sp\in G} C_{S_p},\ G = \{S_1, S_2, S_3, \ldots S_n\} \bigcup_{S_p \in \in G} S_P = G, \\ \forall S_i, S_j \in G \ \text{where} S_i \cap S_j = \phi \tag{4}$$

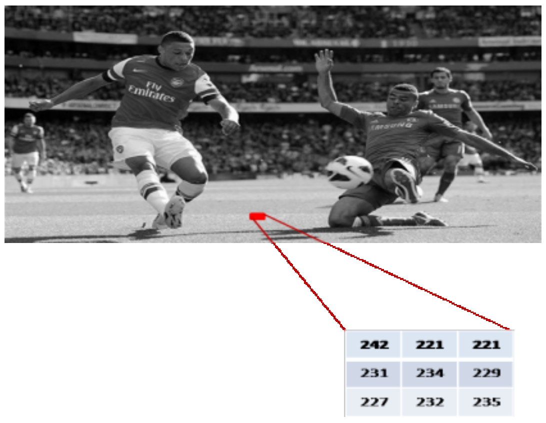


Fig. 2: Pixels of a block from a streaming video frame for prediction

From finding an optimal solution by traversing all possible solutions of complexity 0(nm), the motivation is to solve problem of heterogeneous computing power, hence complex segments has to be mapped to high power computers so that they would comparatively have less number of sub-tasks and their capacity wouldn't be wasted on overhead.

Proposed transcoding algorithm: Here is the motivating scenario where we require discarding of redundancies from the original video. Every heterogeneous device is disseminated with same video content but the only difference is that the quality of the video differs. This is because the quality of each frame and the number of frames received differs. In order to maintain a constant frame quality common to all frames, there should be recovery strategy to maintain a good quality of videos. This recovery strategy should not affect its quality.

This is quite a challenge because we are not only interested in reducing the size of the content but we are keener in maintaining and retaining acute information in our video to be transcoded. When transcoding is performed without proper considerations it may lead to poor quality and playback interruptions. As spotted out in the study of previous papers, the time complexity of such algorithms are large due to the reason that a number of extra factors are computed. We reduce this unnecessary computational overhead by smart coding techniques. We put forth a video transcoding approach based on a predictive approach. We reduce the total number of factors required to only two. The H.264 encoding format has several modes. We choose the sum-of-square errors to be one factor (s). The prediction modes of neighbors of s are determined and predictions from these as chosen as the second factor (s'). We perform smart video transcoding based on s and s'. s' falls on either one of the neighboring sides of s. For instance take s to be the second mode. Hence, s' is the first or third mode. Though there are a few exceptions to this. The 0th or 7th are selected for three, though they are not adjacent neighbors. Similarly for eighth, first and sixth are chosen.

According to existing research, we segregate both the predicted and the reference pixels. To avoid distortions, we consider the original block pixels. The weight value is computed based on the distance between the reference and the predicted values in a block (Fig. 2).

$$S_w = 1/\exp\left\langle \sum_1^n \sum_1^n \mathrm{var} \times \left\{ \frac{\left(0(i,j) - s(i,j)\right)}{n} \right\}^2 \right\rangle \quad (5)$$

The other weight factor is computed as:

$$S'_w = 1/\exp\left\langle \sum_1^n \sum_1^n \mathrm{var} \times \left\{ \frac{\left(0(i,j) - s'(i,j)\right)}{n} \right\}^2 \right\rangle \quad (6)$$

Where n is the size of the block under consideration, var is the variance of the pixel values in s and s'. We sum up the values of these two quantities computed above and use it when normalization is required. Let p denote the block for prediction:

$$p = \frac{s \times s_w + s' \times s'_w}{s_w + s'_w} \quad (7)$$

The variance value differs from block to block. Each time this has to be computed. Hence, we modify this into:

$$S_w = 1/\exp\left\langle \sum_1^n \sum_1^n \mathrm{var} \times \left\{ \frac{\left(0(i,j) - s(i,j)\right)}{\left(0(i,j) - s'(i,j)\right)} \right\}^2 \right\rangle \quad (8)$$

$$S'_w = 1/\exp\left\langle \sum_1^n \sum_1^n \mathrm{var} \times \left\{ \frac{\left(0(i,j) - s'(i,j)\right)}{\left(0(i,j) - s'(i,j)\right)} \right\}^2 \right\rangle \quad (9)$$

Hence, $s'_w = e^{-1}$. The following figure shows the prediction block values of a sample image. Now we have discussed the encoding process. At the decoding end, we have to get back the original image, without any distortion. For this, we need to recover the weight values. We need a process to estimate the weight values from previous blocks. WE make use of a recovery term:

$$z = \frac{s_w}{s_w + s'_w} \quad (10)$$

$$z = \mathrm{sig}(a) \quad (11)$$

where sig(a) is the logistic sigmoid function defined by :

$$f(x) = \frac{1}{1+e^{-x}} \quad (12)$$

Where:

$$\alpha = \frac{\sum_1^n \sum_1^n \left\{ \left(0(i,j) - s'^{(i,j)}\right)^2 - \left(0(i,j) - s'^{(i,j)}\right)^2 \right\}}{\sum_1^n \sum_1^n \left(0(i,j) - s'^{(i,j)}\right)^2} \quad (13)$$

We obtain p'(x,y) as:

$$p' = \frac{z \times s_w + (1-z)s'_w}{s_w + s'_w} \quad (14)$$

Additionally, the original pixel values are also not known at the decoding side. Hence, we have to obtain an optimal expression for o(i,j):

$$0(i,j) = p'(i,j) + s_w(i,j) + s'_w(i,j) + \varepsilon(i,j) \quad (15)$$

The quantity ε is the small amount of additional pixel values which is generated by the algorithm. The value of ε is dependent on three other values namely, ε_{alg}. ε_{sw} and $\varepsilon_{s'w}$. ε_{alg} is the additional pixel values generated by the algorithm. ε_{Sw} and $\varepsilon_{s'w}$ are generated by s_w and s'_w, respectively. The relationship between these three parameters can be represented in the matrix format as:

$$\begin{pmatrix} \varepsilon_{sw} \\ \varepsilon_{s'w} \end{pmatrix} = \begin{pmatrix} 1-z & 0 \\ 0 & -z \end{pmatrix} \begin{pmatrix} s'_w(i,j) & -s_w(i,j) \\ s'_w(i,j) & -s_w(i,j) \end{pmatrix} + \begin{pmatrix} \varepsilon_{alg} \\ \varepsilon_{alg} \end{pmatrix} \quad (16)$$

We compute the deviation between S_w and S'_w. This value is used to reformulate the expression for z in terms of another parameter 'b'. The deviation between the two quantities is also a matrix represented by **δ**. Let **δ** (i, j) represent a particular pixel of the deviation matrix. Let Δ (I, j) represent the value of **δ** $(i, j)^2$ for a particular pixel value:

$$b = \sum_1^n \sum_1^n \frac{\Delta(i,j) + \varepsilon_{alg} - 2\left[z\Delta(i,j) + \Delta(i,j)\right]}{z^2\Delta(i,j) + \varepsilon_{alg} + \left(\varepsilon_{alg}\right)^2 - 2z\Delta(i,j)} \quad (17)$$

$$z = sig(b) \quad (18)$$

Algorithm 1: Proposed EMPT transcoding algorithm

```
Step 1: The overall algorithm for the proposed approach is as follows:
Step 2: Choose $s_w$ optimally
Step 3: Based on s value, compute s' as adjacent. (Pay attention to special cases).
Step 4: Choose the minimal value of s' from the two obtained.
Step 5: For encoding, estimate the weight values - $s_w$ and $s'_w$
Step 6: At the decoding end, the original block is recovered using b and z by reformulating the weight values.
```

Proposed Optimal Reduced Finish Time (ORFT): The second proposed work of this study is a novel algorithm which is Optimal Reduced Finish Time (ORFT) aiming to reduce the finish time of the complete transcoding process. This algorithm works with two procedures: Knapsack, re-assignment procedure. In Knapsack procedure, the lower bound of the finish time is calculated and complex sub tasks are assigned to powerful machine checking its overflow condition and in the re-assignment procedure, the residual segments are crosschecked with machines to be accommodated so that the total time taken for transcoding the previously assigned sub-tasks and the currently assigned residual segment should be lesser than the average finish time.

Knapsack procedure: Initially there are "m" computers and "n" blocks, hence the average finish time is estimated and this value is taken as the lower bound:

$$\hat{C} = {}^{\sum c_b}\!\Big/\!{}_{\sum M_p} + J_{overhead} \times {}^{n}\!/\!{}_{m} \quad (19)$$

Where, C^ is the average complete time and taking this as the lower bound the solution will be obviously optimal since there would be few machined whose finish time might exceed the lower bound. Here the machines are considered as a Knapsack and we specify an overflow condition. The volume of each machine is calculated as:

$$V_i = p_i \times \hat{C} \quad (20)$$

We assume CP as the complete time for the machine 'm1' with power "p1" for the previously assigned sub-task. Before assigning the residual segment "r" on the machine "m1" overflow condition must be ensured and calculated as:

$$C_p + t_p^r > \hat{C} \quad (21)$$

If the complete time along with the transcoding time of residual segment is greater than the average complete time, then the machine would overflow and the residue should be accommodated in any other machine whose complete time would be much less than the average complete time.

Re-assignment procedure: After the Knapsack procedure, there will always be some space in every machine between the actual complete time "CP" and the average complete time "C" hence, the segments has to be assigned in such a way so as to reduce the overall finish time. In re-assignment procedure traversing begins in descending order. Hence, the machine with minimal complete time is chosen and the residual segment is assigned to that particular machine:

$$t^{r}_{bp} = C \;\; + t_{bp} \qquad (22)$$

Algorithm 2: Proposed ORFT algorithm

```
Input: Video segments of „n" blocks B = { $b_1$, $b_2$, $b_3$,.....$b_n$} with varying complexities C = { $c_1$, $c_2$, $c_3$,.....$c_n$}.
Output: "n" blocks of transcoded video segments.
Step 1: Arrange the blocks B in an order of descending complexity suchthat ($c_1>c_2>c_3 ....... >c_n$)
Step 2: Arrange 'm' computers M = { $p_1$, $p_2$, $p_3$, ......$p_m$} with different power are also in an order of descending capacity such that($P_1>P_2>P_3 .... P_m$)
Step 3: Compute the transcoding time for a block 'b' on a computer with power 'p' as:
    $t_{bp} = C / M_p + J_{overhead}$
Step 4: Calculate the completion time for set of segments $S_p$ as:
    $C_{Sp} = \sum_{b\in Sp} t_{bp} = \sum_{b\in Sp} C_b / M_p + |S_P| \times J_{overhead}$
Step 5: Mapping of segments to computers
    • Knapsack Procedure. Average Finish time is computed as:
        $\hat{C} = \sum C_b / \sum M_p + J_{overhead} \times n/m$
      Calculate the volume of each machine as:
        $V_i = p_i \times \hat{C}$
      Overflow condition is calculated as:
        $C_p + t^{r}_{p} > \hat{C}$
    • Re-assignment procedure. Calculate the finish time of block 'b' on each computer as:
        $t^{r}_{bp} = C_p + t_{bp}$
```

Algorithm analysis: This study describes the complexity of the proposed ORFT algorithm. The overall complexity of the algorithm is found out by considering Knapsack complexity o(n), re-assignment procedure with complexity o(nm) and last segment sorting with complexity o(n log n). Hence, the complexity of the proposed algorithm has a low complexity of O(n log n).

Here we assume the number of blocks "n" is greater than the number of machines "m". If "m" is greater than "n", then it is easy to prove. It is denoted as:

$$C_{gross} = \max_{S_p \in G} C_{S_p}$$

Theorem 1: To prove the gross complete time obtained by the proposed ORFT algorithm is not greater than double of the average complete time, that is:

$$C_{gross} \le 2 \times C\hat{} = 2\left(\sum C_b \Big/ \sum M_p + J_{overhead} \times n/m \right)$$

Proof: Lets assume that the gross complete time obtained by the proposed ORFT algorithm is greater than double of the average complete time. In such a case „n" would be small and the segment with least complexity will be the last to be scheduled as well as to be completed. We can prove it but taking the smallest segment as 'n' and C_{gross} is the complete time of the computer which handles 'n'. Then where c_n is the segment with least complexity:

$$C_{gross}\, p_i \le C_p\, p_i + C_n p_i J_{overhead}$$

$$C_{gross} \sum p_i \le \sum C_p p_i + mc_n + J_{overhead} \sum p_i$$

$$C_{gross} \le \sum C_p p_i \Big/ \sum p_i + mc_n \Big/ \sum p_i + J_{overhead}$$

$$C_{gross} \le \sum C_b \Big/ \sum M_p + n \times J_{overhead} \big/ m + mc_n \Big/ \sum p_i + J_{overhead}$$

$$\sum C_b \Big/ \sum M_p + n \times J_{overhead} \big/ m < mc_n \Big/ \sum p_i + J_{overhead}$$

It is said that cn is the segment with least complexity, n<m. The above expression doesn't get through and obviously it is an incorrect assumption. Hence theorem is proved.

MATERIALS AND METHODS

As stated earlier, this paper has concentrated on task scheduling to ensure fast transcoding using cloud based MapReduce. To implement the proposed algorithm, video size, length, power of the machine and job assigning overhead is considered. The work has been implemented in Matlab and experiments were conducted and results were evaluated for different scheduling strategies.

Hadoop 2.6.0 was deployed using Metal as a service and juju. The reason for using juju is because it provides a convenience to modify the services being deployed in real time. Here, MapReduce 2.0 (MRv2) or YARN is used for performing the proposed scheduling strategy, based on the chunk size Vm's are allocated, jobs are executed and final transcoded video is reduced and given out.

For our experimental set up 10 machines were created and their processing capacity and volume were ranged between 5-15. About 300 of video segments were used and their complexities ranged from 300-900 approximately. The job assigning overhead was fixed to 7. For every strategy 500 experiments were carried out and the average was picked for evaluation purpose. In our experiments evaluations were compared for varying number of video segments, varying the machine power and the finish time taken for transcoding the video segments with and without similar frames.

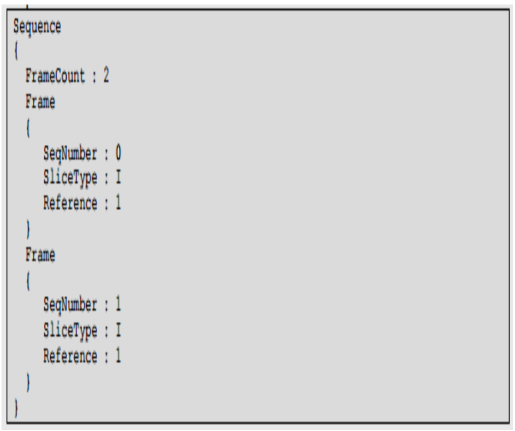
At the receiving end, we have used 60 Acer Veriton X4630G personal computer each with the configuration of 64 bit Windows 7 operating system architecture, i5 processor, 4 GB RAM and 500 GB of hard drive capacity, one Dell precision 690 workstation of specification quad-core Intel Xeon 5300 series processor with upto 1333 MHz front side bus and 2x 4GB shared cache, upto 32 GB, mobile phones of brands such as Nokia with the screen resolution of 480x800 and 240x320, Blackberry with 320x240 and Micromax with 480x854 along with various specifications were used as heterogeneous devices. These devices were connected as peer-to-peer with physical network strength of 100 Mbps. Some devices were switched off and switched on over the fly in order to maintain the dynamic nature. With the devices that are active, a tree based overlay was constructed which was evaluated by joining and removing the nodes from the network.

RESULTS AND DISCUSSION

First the results of our proposed transcoding algorithm is illustrated.. One of the widely used metrics used to determine the amount of misfit from an image or a signal transmitted from a source to the one that is received at the receiver is the (Peak Signal-to-Noise) PSNR metric. This is more prevalent in video processing evaluations too. Hence, we also adopt the same metric to evaluate the performance and efficiency of our algorithm. Let I and I' represent the image at source and image at the receiving end respectively. The PSNR value is computed as:

$$\text{PSNR} = 10\log_{10}\left\{\frac{65025}{\sum_1^n \sum_1^n \left[I(i,j) - I'(i,j)\right]^2 / n^2}\right\}$$

This is valid for a square image of size n×n. In case where the height and the width of the image are not equal, we represent the PSNR value as:



```
Sequence
{
  FrameCount : 2
  Frame
  {
    SeqNumber : 0
    SliceType : I
    Reference : 1
  }
  Frame
  {
    SeqNumber : 1
    SliceType : I
    Reference : 1
  }
}
```

Fig. 3: JM parameters

$$\text{PSNR} = 10\log_{10}\left\{\frac{65025}{\sum_1^m \sum_1^n \left[I(i,j) - I'(i,j)\right]^2 / mn}\right\}$$

We have implemented the algorithm and tested it against H.264 AVC standard of MPEG-4 video segments. The encoder parameters are used as defined in the defines.h and configfile.h. The internal file format structure of the H.264 format file is as follows (Fig. 3).

The experiment was carried out using JM. The parameters of the JM modules are set accordingly. For example, the number of frames, frame rate, rate control are set. We have set the number of frames to be 250 and the frame rate to be 60 fps. The Quantization Parameter (QP) is set between 30 to 35. We compare our results to the predefined JM settings. This is tabulated as follows (Table 1).

All the videos listed in the above table are our own custom videos. From Table 1, it is clear that our approach is superior to the existing approach by producing around 6-17% lesser bit rate. And the PSNR value is by 0.4-0.7. This signifies that the image I' upon receiving is more closer to the source image I than previous approaches.

This is achieved by the recovery and reformulation of the weights step in our algorithm. Next we show the comparison results between the encoding times of JM default and existing approach our proposed algorithm. The percentage of increase in the encoding time can be represented as:

$$\%\ \text{increase} = \left(\frac{\text{Time of proposed}}{\text{JM encoding time}} \times 100\right)$$

Table 1: PSNR and bit rate comparison

		Existing		Proposed	
Video	Resolution	PSNR (dB)	Bit rate (%)	PSNR (dB)	Bit rate (%)
Video1	352×288	0.17	6.78	0.399	12.02
Video2	352×288	0.32	5.45	0.44	8.47
Video3	352×288	0.58	5.04	0.69	7.76
Video4	352×288	0.49	4.67	0.63	6.22
Video5	176×144	0.36	7.92	0.50	17.35
Video6	176×144	0.27	6.17	0.41	11.85
Video7	176×144	0.43	6.81	0.57	12.26

Table 2: Comparison of Encoding times and % increase

		Existing		Proposed	
Video	Resolution	Encoding time (ms)	Increase (%)	Encoding time (ms)	Increase (%)
Vid 1	352x288	2117	157	1745	129
Vid 2	352x288	3014	162	2798	150
Vid 3	352x288	1752	153	1377	120
Vid4	352x288	2067	145	1689	118
Vid5	176x144	1298	154	1036	122
Vid6	176x144	1045	166	948	150
Vid7	176x144	1173	158	997	134

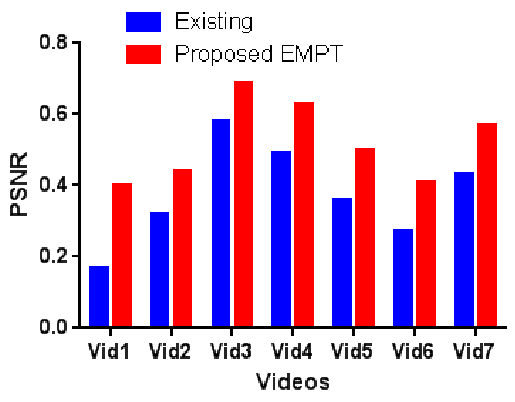


Fig. 4: Comparison rate PSNR

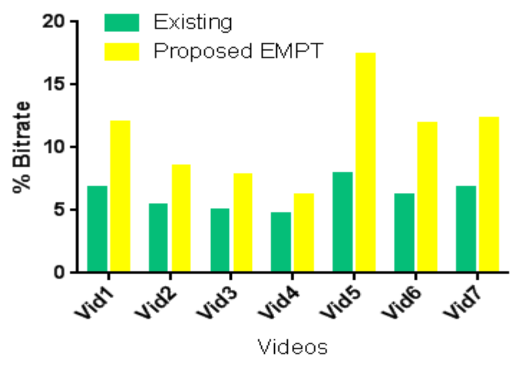


Fig. 5: Comparison of bitrates

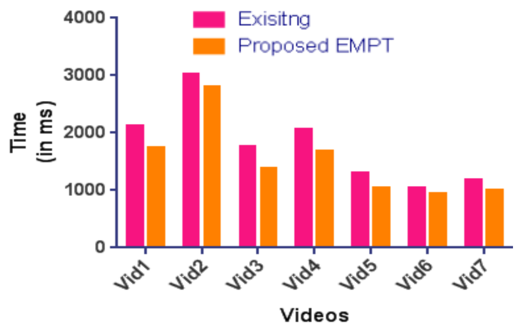


Fig. 6: Comparison of Encoding times (in ms)

It is found on the average that, the percentage of increase in the proposed approach is very less than the existing approach. Therefore, our algorithm has a better transcoding performance than the existing one (Table 2). In Fig. 4 the video segments are varied, the differences in the PSNR values also varies. The proposed EMPT algorithm produces high PSNR output than the existing system.

Figure 5 shows the plot of the comparison between the bitrate of the various videos in terms of percentage for the existing and proposed systems. The bitrates of the various videos are found to be higher in the proposed EMPT algorithm compared to the existing approach. Figure 6 shows that the proposed EMPT algorithm has lesser encoding times for various videos when compared to the existing approach.

Figure 7 shows the comparison of the encoding times of both the existing and proposed approaches with the standard H.264 AVC encoding time. Figure 8 shows the percentage increase in the existing and proposed approaches. It is found that the proposed approach out performs the existing approach. Figure 9 shows the comparison of the percentage increase in coding times

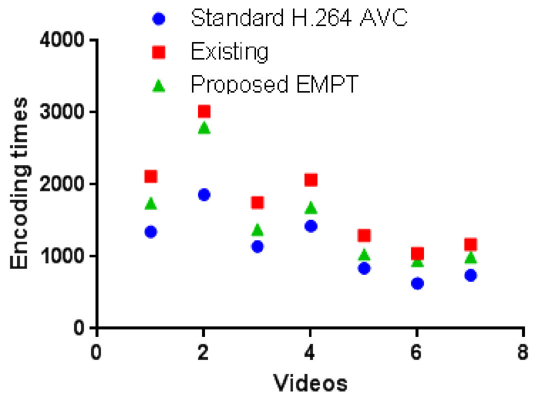


Fig. 7: Comparison of encoding times with standard H.264 AVC

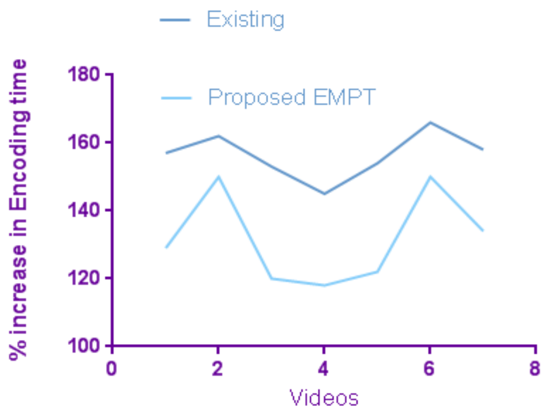


Fig. 8: Percentage increase in the encoding time

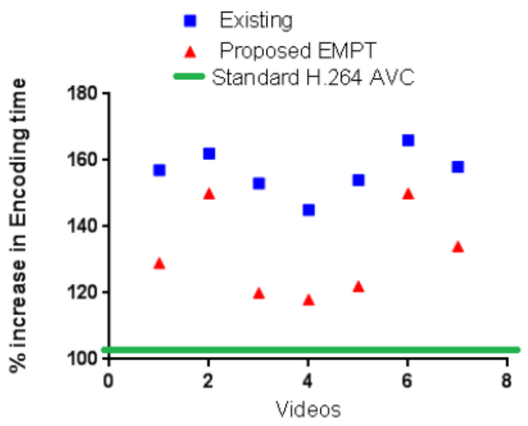


Fig. 9: Percentage increase comparison

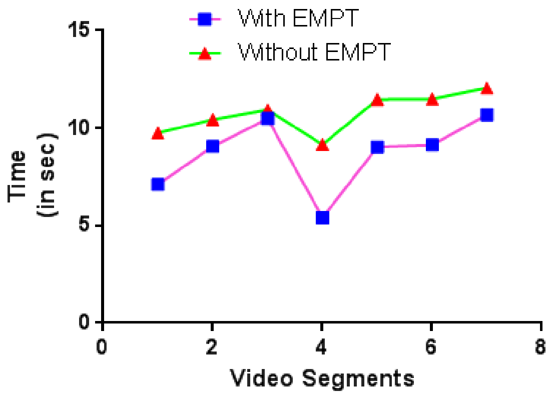


Fig. 10: Completion time with and without EMPT

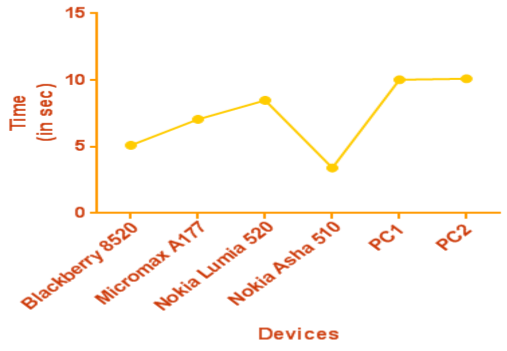


Fig. 11: End-to-end in various devices

with the standard H.264 AVC base time. Figure 10 shows the completion time using the proposed ORFT algorithm with and without enhancing it with similar frame elimination algorithm. Figure 11 shows the end-to-end delay incurred in various heterogeneous devices represented in seconds.

CONCLUSION

Cloud based transcoding system was designed to stream live contents. This study has dealt with two proposed work one is the proposed transcoding algorithm and the next is ORFT algorithm. We have proposed an efficient transcoding technique based on predictions on texture analysis. We have also reduced the number of modes used in the prediction process without affecting the efficiency of prediction. The increase in the encoding time is also lesser than the existing approaches and the achieved PSNR ratio is lesser. Thus this transcoding scheme can be efficiently used in applications like live streaming. Scheduling was carried out considering different complex video segments, machines with varying power and capacity and job assigning overhead. To reduce the complexity of transcoding operations and ensure fast transcoding Optimal Reduced Finish Time (ORFT) algorithm is proposed. This scheduling strategy results in better allocation of chunks to the perfect matching machine based on its computing power. Mathematical simulations and algorithm analysis proves that the proposed MapReduce based transcoding model performs better than other transcoding algorithms.

REFERENCES

Balsree, R., A. Thawani, S. Gopalan and V. Sridhar, 2005. Inter-frame similarity based video transcoding. Proceedings of the 7th IEEE International Symposium on Multimedia (ISM'05), December 12-14, 2005, IEEE, Bangalore, India, ISBN: 0-7695-2489-3, pp: 1-6.

Cha, M., H. Kwak, P. Rodriguez, Y.Y. Ahn and S. Moon, 2007. I tube you tube everybody tubes: Analyzing the worlds largest user generated content video system. Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, October 23-26, 2007, ACM, San Diego, California, ISBN: 978-1-59593-908-1, pp: 1-14.

Evangeline, D.P., P. Anandhakumar, 2015. SEASONS-A scalable 4-tier cloud architecture for handling heterogeneity of mobile devices in live multimedia streaming. ARPN. J. Eng. Appl. Sci., 10: 2017-2022.

Han, S., M. Kim, Y. Cui, S. Seo, S. Seo and H. Lee, 2014. CMTS: Cloud multimedia transcoding system. Int. J. Adv. Comput. Networking, 2: 21-23.

Huang, Z., C. Mei, L.E. Li and T. Woo, 2011. Cloud Stream: Delivering high-quality streaming videos through a cloud-based SVC proxy. Proceedings 2011 IEEE Conference on INFOCOM, April 10-15, 2011, IEEE, Shanghai, China, ISBN: 978-1-4244-9919-9, pp: 201-205.

Liu, Y., F. Li, L. Guo, B. Shen and S. Chen et al., 2013b. Measurement and analysis of an internet streaming service to mobile devices. IEEE. Trans. Paral. Distrib. Syst., 24: 2240-2250.

Liu, Y., F. Li, L. Guo, B. Shen and S. Chen, 2013a. Effectively minimizing redundant Internet streaming traffic to IOS devices. Proceedings of the 2013 IEEE Confrence on INFOCOM, April 14-19, 2013, IEEE, Turin, Italy, ISBN: 978-1-4673-5944-3, pp: 250-254.

Liu, Y., F. Li, L. Guo, B. Shen and S. Chen, 2012. A servers perspective of internet streaming delivery to mobile devices.q Proceedings 2012 IEEE Conference on INFOCOM, March 25-30, 2012, IEEE, Orlando, Florida, ISBN: 978-1-4673-0773-4, pp: 1332-1340.

Liu, Y., F. Li, L. Guo, Y. Guo and S. Chen, 2011. Bluestreaming: towards power-efficient internet p2p streaming to mobile devices. Proceedings of the 19th ACM International Conference on Multimedia, November 28-December 01, 2011, ACM, Scottsdale, Arizona, ISBN: 978-1-4503-0616-4, pp: 193-202.

Sullivan, G.J., J.R. Ohm, W.J. Han and T. Wiegand, 2012. Overview of the high efficiency video coding standard. IEEE. Trans. Circuits Syst. Video Technol., 22: 1649-1668.

Yang, C.C., G.L. Li, M.C. Chi, M.J. Chen and C.H. Yeh, 2010. Prediction error prioritizing strategy for fast normalized partial distortion motion estimation algorithm. IEEE. Trans. Circuits Syst. Video Technol., 20: 1150-1155.

Zhu, Z., S. Li and X. Chen, 2013. Design QoS-aware multi-path provisioning strategies for efficient cloud-assisted SVC video streaming to heterogeneous clients. IEEE. Trans. Multimedia, 15: 758-768.