ISSN:0975-9646

Client-Side Scheduling For Video-Streaming Using Clouds

T Venkatesh[1], Santhana G[2], Swathi L S[2], Sushmashree S[2], Arpitha C[2]

[1]*Associate Professor, Dept of C.S.E, Ghousia College of Engineering, Ramanagaram.*

[2]*Students of 8th semester, Computer Science and Engineering, Ghousia College of Engineering, Ramanagaram.*

Abstract-Long buffering time and intermittent disruptions are caused by increasing traffic demands and time-varying link capacity over wireless networks which results in poor QoS in videos. Using Cloud technology we propose a model for efficient video streaming using a layered-approach namely Scalable Video Coding (SVC). By obtaining the transmission status of each client periodically, adaptive streaming is provided by the private agents in the clouds effectively. In this paper, we present the idea of scheduling the videos at the client side by providing users with their choice of scheduling mechanisms. Here we consider three mechanisms namely First Come First Serve (FCFS), Shortest Job First (SJF), Priority Scheduling and compare their throughput and performance.

Keywords : Scalability ; Adaptability ; SVC ; Scheduling ; Cloud.

I. INTRODUCTION

Video coding today is used in a wide range of applications ranging from multimedia messaging, video telephony and video conferencing over mobile TV, wireless and Internet video streaming, to standard- and high-definition TV broadcasting. In particular, the Internet and wireless networks gain more and more importance for video applications.

The variation of wireless channel capacity and tight delay constraints make the delivery of video difficult. In this paper we proposed a performance model for real-time video transmission over the uplink of the third generation wireless network.

The importance of a Variable Bit Rate (VBR) video transmission on wireless networks is increasing in time. The bursty nature of VBR traffic complicates the design of efficient mechanisms for video retrieval, transport, and provisioning to achieve high bandwidth utilization and reduce the negative effects of bandwidth fluctuations in wireless channels. To this aim, several scheduling algorithms can be successfully implemented. They regulate data transmission to reduce the rate variability peculiar of VBR streams. At client side, scheduled data are temporarily stored in the client buffer before being decoded on the terminal. The suggested algorithm thought for VBR stream transmission in wireless networks that takes into account the user interactivity. Scheduling is performed over relatively small video segments to reduce delays. We implement and compare the results of three traditional algorithms: FCFS (First Come First Serve), SJF (Shortest Job First) and Priority scheduling algorithm.

The remaining of the paper is organized as follows: Section II deals with Related Work, Section III emphasizes on Proposed System which is subdivided into two sections – A. Scalable Video Coding (SVC) and B. Scheduling Approaches, Section IV presents the Experimental Results; Section V concludes our views on this paper.

II. RELATED WORK

Although adaptive transmission strategies, such as adaptive video data scheduling, can be employed, deriving the optimal adaptive transmission policy is difficult because the transmission strategies taken at different time are coupled with each other via receiver buffer state. Furthermore, due to the nature of predictive video coding algorithms, a video frame can be decoded only when its predictors have been received. Hence, the prediction structure of the video codec enforces a partial order on the transmissions of the video packets, which limits the flexibility of adaptive video transmission.

Scheduling is carried out to reduce the video traffic at the client end. We adopt the CPU scheduling approaches for videos based on different criteria to enhance the performance to the fullest.

III. PROPOSED SYSTEM

Recently there have been many studies on how to improve the service quality of mobile video streaming on two aspects:

A. Scalability - the available link capacity of a mobile device may vary over time and space depending on its signal strength, other user's traffic in the same cell, and link condition variation. Storing multiple versions (with different bit rates) of the same video content may incur high overhead in terms of storage and communication. To address this issue, the Scalable Video Coding (SVC) technique of the H.264 AVC video compression standard defines a base layer (BL) with multiple enhance layers (ELs).

B. Adaptability – here we have to adjust the video bit rate adapting to the currently time-varying available link bandwidth of each user.

However most of the proposals seeking to jointly utilize the video scalability and adaptability rely on the active control on the server side. That is, every mobile user needs to individually report the transmission status (e.g., packet loss, delay and signal quality) periodically to the server, which predicts the available bandwidth for each user. Thus the problem is that the server should take over the

substantial processing overhead, as the number of users increases.

Cloud computing techniques are poised to flexibly provide scalable resources to content/service providers, and process offloading to mobile users.

Our framework uses SVC for effective video streaming in the network, that is, a single video in cloud when fetched by multiple users is delivered to each user without disruptions. Similarly at the client-side, when multiple videos are received by a particular user concurrently, we implement the concept of scheduling at client machine to avoid congestion.

The following two sections discuss about Scalable Video Coding (SVC) and Scheduling Approaches.

A. Scalable Video Coding (SVC)

This technology allows each video stream to be split into a base layer for providing a basic video image and multiple enhancement layers for enhancing video quality. Instead of allocating bandwidth (i.e., tiles) to an entire video stream, SVC flexibility effectively improves bandwidth efficiency.

To view a video, users must receive the base layer. When users further receive the enhancement layers, the video quality improves layer by layer. Then the enhancement layer helps only when users already receive all the n-1 lower layers. In the current systems, users require a long time to decode/encode a video stream; for example, using the Windows Media Encoder requires approximately 10 s. To play a smooth video, users typically buffer several video streams before displaying video images. Therefore, video streaming services are generally delay-tolerant.

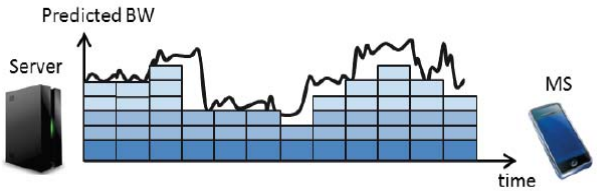

Figure 1 : SVC layers of BL and ELs

Scalable video coding (SVC) is one approach to enable flexible video transmission over channels with varying throughput. An SVC video encoder produces a layered video stream that contains a base layer and several enhancement layers. If the throughput is low, the transmitter can choose to transmit the base layer only, which provides a moderate, but acceptable, degree of visual quality at the receiver. If the channel conditions improve, the transmitter can transmit one, or more, enhancement layers to further improve the visual quality. Conceptually, SVC provides a means to adapt the data rate for wireless video transmission. The wireless transmitter can adapt the data rate by selectively scheduling video data associated with various layers for transmission rather than transcoding the video sequence into a different rate.

Designing scalable video scheduling algorithms for wireless channels is a complex task. The scheduling policy depends not only on the channel conditions, but also on the receiver buffer state. For example, if the receiver has successfully buffered base layer data over many frames, the scheduler could choose to transmit some enhancement layer data to improve the video quality even if the throughput is low. At any time, the scheduling decision will determine the receiver buffer state which, in turn, affects the future scheduling decisions. Therefore, adaptive video data scheduling is a sequential decision problem.

SVC encodes video into “layers,” starting with the “base” layer, which contains the lowest level of detail spatially (resolution), temporally (frames per second) and from a quality perspective (higher detail). Additional layers can increase the quality of the stream using any or all of these variables.

For example, the base layer of a stream might be encoded at 15 frames per second, a resolution of 320x240, and a data rate of 300 kbps. Additional layers could expand that stream to 720p video at 3 mbps suitable for a set top box, with convenient stopping points for relative high quality streaming over the Internet, say at 640x480x30 fps @ 600 kbps and 720p at 2 mbps.

SVC enables the transmission and decoding of partial bit streams to provide video services with lower temporal or spatial resolutions or reduced fidelity while retaining a reconstruction quality that is high relative to the rate of the partial bit streams. Hence, SVC provides functionalities such as graceful degradation in lossy transmission environments as well as bit rate, format, and power adaptation. These functionalities provide enhancements to transmission and storage applications. SVC has achieved significant improvements in coding efficiency with an increased degree of supported scalability relative to the scalable profiles of prior video coding standards.

The Scalable Video Coding amendment (SVC) of the H.264/AVC standard (H.264/AVC) provides network-friendly scalability at a bit stream level with a moderate increase in decoder complexity relative to single-layer H.264/AVC. It supports functionalities such as bit rate, format, and power adaptation, graceful degradation in lossy transmission environments as well as lossless rewriting of quality-scalable SVC bit streams to single-layer H.264/AVC bit streams. These functionalities provide enhancements to transmission and storage applications. SVC has achieved significant improvements in coding efficiency with an increased degree of supported scalability relative to the scalable profiles of prior video coding standards.

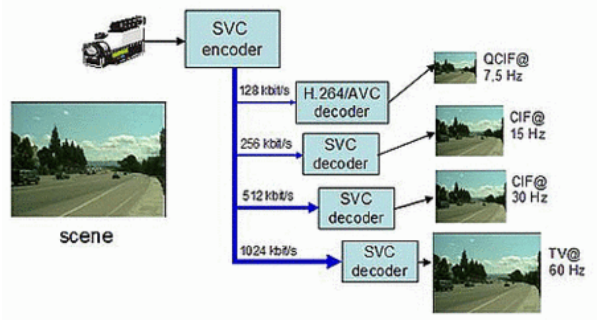

Figure 2: Scalable and Adaptable feature of SVC

A video bit stream is called scalable when parts of the stream can be removed in a way that the resulting sub-stream forms another valid bit stream for some target decoder, and the sub-stream represents the source content with a reconstruction quality that is less than that of the complete original bit stream but is high when considering the lower quantity of remaining data. Bit streams that do not provide this property are referred to as single-layer bit streams. The usual modes of scalability are temporal, spatial, and quality scalability. Spatial scalability and temporal scalability describe cases in which subsets of the bit stream represent the source content with a reduced picture size (spatial resolution) or frame rate (temporal resolution), respectively. With quality scalability, the sub-stream provides the same spatio-temporal resolution as the complete bit stream, but with a lower fidelity – where fidelity is often informally referred to as signal-to-noise ratio (SNR). Quality scalability is also commonly referred to as fidelity or SNR scalability. More rarely required scalability modes are region-of-interest (ROI) and object-based scalability, in which the sub-streams typically represent spatially contiguous regions of the original picture area. The different types of scalability can also be combined, so that a multitude of representations with different spatio-temporal resolutions and bit rates can be supported within a single scalable bit stream.

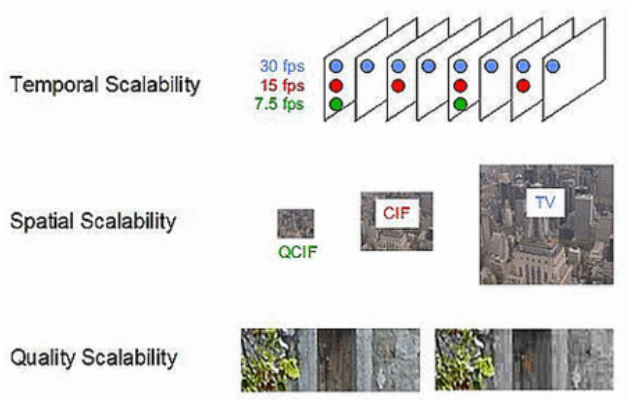

Figure 3: Scalability in SVC

a) Temporal scalability - A bit stream provides temporal scalability when the set of corresponding access units can be partitioned into a temporal base layer and one or more temporal enhancement layers with the following property. Let the temporal layers be identified by a temporal layer identifier T, which starts from 0 for the base layer and is increased by 1 from one temporal layer to the next. Then for each natural number k, the bit stream that is obtained by removing all access units of all temporal layers with a temporal layer identifier T greater than k forms another valid bit stream for the given decoder.

b) Spatial scalability - For supporting spatial scalable coding, SVC follows the conventional approach of multi-layer coding, which is also used in H.262/MPEG-2 Video, H.263, and MPEG-4 Visual. In each spatial layer, motion-compensated prediction and intra prediction are employed as for single-layer coding. In addition to these basic coding tools of H.264/AVC, SVC provides so-called inter-layer prediction methods, which allow an exploitation of the statistical dependencies between different layers for improving the coding efficiency (reducing the bit rate) of enhancement layers

c) Quality scalability - Quality scalability can be considered as a special case of spatial scalability with identical picture sizes for base and enhancement layer. This case, which is also referred to as coarse-grain quality scalable coding (CGS), is supported by the general concept for spatial scalable coding. The same inter-layer prediction mechanisms are employed, but without using the corresponding upsampling operations. When utilizing inter-layer prediction, a refinement of texture information is typically achieved by re-quantizing the residual texture signal in the enhancement layer with a smaller quantization step size relative to that used for the preceding CGS layer.

Compared to other approaches, H.264 SVC is very efficient, as the SVC encoded file should only be about 20% larger than the file size necessary to supply the highest quality representation. In other words, if you were encoding the file to send to the set top box at 3 mbps, the SVC encoded file would have an overall data rate of about 3.6 mbps. In addition, the base layer should be compatible with existing H.264 players, so no player upgrade will be necessary to view the base layer stream. With hardware encoders, streaming producers can convert their current formats to SVC compatible streams on the fly, so video publishers like CNN and ESPN won't need to convert their entire library to leverage the new technology.

B. Scheduling Approaches

Scheduling is one of the most important tasks of operating system, which provides a specific sequence of execution for the jobs waiting in queue. The scheduling of jobs has very much impact on efficiency and performance of CPU. The major objective of this task is to maximize the efficiency and hence the performance.

CPU scheduling means allocating CPU time to the processes waiting in ready queue. Scheduling should be done fairly and correctly, so that every process get chance to execute on processor. CPU scheduling can be done based on various criteria such as waiting time, average waiting time, turnaround time, average turnaround time etc. There are number of CPU scheduling algorithms available like First Come First Serve (FCFS), Shortest Job First (SJF), etc.

A scheduling algorithm is the method by which tasks, processes, threads or data flow are given access to system resources. Generally a set of criteria is established against which various scheduling policies may be evaluated.

- CPU Utilization - In a multi programmed operating system, CPU should be as busy as possible so has to execute more jobs.
- Turnaround Time - This is the sum of the periods spent waiting to get into memory, waiting in the ready queue, executing on the CPU and doing I/O.
- Waiting Time - This is the sum of the periods spent waiting in the ready queue.
- Response Time - This is the time from the submission of a request until response begins to be received.

Scheduling strategies can be broadly divided into two classes: *non-preemptive* strategies and *preemptive* strategies. In case of preemptive scheduling, the processor switches from one task to another task by considering various factors. On the other hand in non-preemptive scheduling once the processor is allocated with a specific task it doesn't service any other incoming task until the completion of currently processing task. Since we are dealing with videos specifically, preemption in videos would result in user inconvenience hence we consider only non-preemptive strategy in our paper.

The literature has presented various scheduling algorithms aimed at allocating bandwidth efficiently. Most prominently used algorithms are First Come First Serve (FCFS), Shortest Job First (SJF), Priority.

Scheduling is a complex challenge in cloud computing. In traditional distributed environment, the aim of optimizing scheduling is mainly focusing on system performance, such as system throughput, CPU utilization rate and almost never considering QoS. In cloud computing environment, we are not only emphasizing resource utilization rate and system performance, but also requiring a guaranteed QoS of users based on different demands.

The major factors that decide the effective scheduler is to reduce the cost under various circumstances. Cloud computing is a non-static environment it difficult to maintain a load balance in the system. Cloud Service provider more concentrates on to reduce the gap between the over-provision of resources during the peak time and the under provision of resources at non-peak time.

Most fit task scheduling algorithm select the best fit job executed first, failure to select optional job. The proposed approach enhances the scheduler grouping the various burst time based jobs into queue.

a) FCFS - First Come First Serve (FCFS) is one of the simplest algorithms for job scheduling. It is just a FIFO queue of processes waiting for their turn, similar to the queue at bus stop or at ticket counter. It simply selects the first arrived process for execution from the ready queue.

FCFS schedules jobs in the order of their arrival into the job queue. In the First Come First Serve job scheduling the jobs are queued in the order of which come first.

In FCFS scheduling job arrival is fair and predictable, but the drawback is starvation, leading to low utilization, that is, if process with longest burst time is executed first, then the process with shortest burst time will wait unnecessarily for long duration.

b) SJF - Shortest Job First (SJF) is one of the efficient CPU scheduling techniques. This strategy takes job service times into consideration in making scheduling decisions. Each time a process with shortest burst time is selected for execution from the processes in ready queue. This ensures the minimum average waiting time and also overcomes the drawback of FCFS.

In Shortest Job First they give more priority to small jobs, medium and long jobs are executed after the execution of small jobs. The client submitted jobs sorted based on the ascending order of the burst time and it gives equal importance to all jobs. It helps to increase the client satisfaction because the client requirements are varying based on the current needs. It reduces starvation using the dynamic allocation of jobs to select the best suitable jobs from among the available and does not decrease the performance of the system.

The SJF policy reduces the job response times substantially (as compared to the FCFS scheduling policy). The limitations of SJF are: it requires future knowledge about the CPU burst time and it will penalize the process with longest burst time.

c) Priority - Priority job scheduling algorithm each job is executed based on the priority. The priority assignment reflects the message urgency. Intuitively, the smaller the relative deadline of a connection is, the higher its priority should be.

The low priority queue is served only if the high priority one is empty. When a video of the high priority is set in its queue, if a video of the low priority flow is being sent, it must wait until the end of this video before being serviced (this is a non preemptive scheduling).

IV. EXPERIMENTAL RESULTS

We consider sample inputs for burst time and evaluate the Average Turn Around Time (ATAT) using different scheduling mechanisms. This is depicted in Table 1. Some of the observations that can be inferred from the table are as follows:

- For input burst time (6, 8, 7, 3), we obtain ATAT of 16.25 for FCFS and Priority, but a minimal of ATAT is obtained for SJF. With this, we could say that for smaller differences in Burst time of the tasks, there is little difference in ATAT among FCFS and Priority. When compared to other scheduling schemas, SJF yields the optimal solution.
- For larger differences in burst time, a lot of variation occurs. We consider (18, 6, 24, 4) as an example input to illustrate this scenario. By assuming that the first task has the maximum priority, the output of FCFS and Priority is exactly same.
- When the task having the largest burst time among the set of tasks is selected to have the highest priority, priority scheduling proves to produce an inefficient result compared to FCFS.
- When the task having the smallest burst time among the set of tasks is selected to have the highest priority, priority scheduling proves to produce an more efficient result compared to FCFS.
- Finally, SJF produces an optimal ATAT compared to any other methods, but the only disadvantage is that the task having largest burst time have to wait for a longest time, that is, it has to starve for a long time.

The comparison of various scheduling schema results of Average Waiting Time and Average Turn Around Time is presented in the graphical form in Figure 4.

Burst time	First Come First Serve	Shortest Job First	Priority
(6,8,7,3) Highest priority=task3	16.25	13	16.25
(18,6,24,4) Highest priority=task2	35.5	23.5	32.5
(18,6,24,4) Highest priority=task1	35.5	23.5	35.5
(18,6,24,4) Highest priority=task3	35.5	23.5	41.5
(18,6,24,4) Highest priority=task4	35.5	23.5	26.5

Table 1 : Evaluation of Average Turn Around Time for various scheduling mechanisms

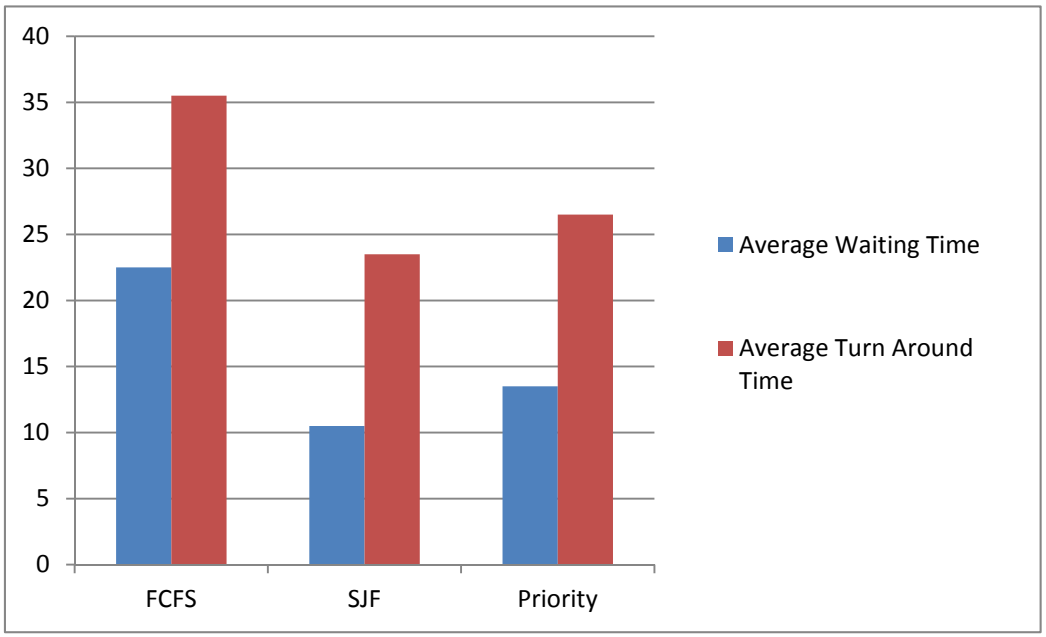

Figure 4 : Graphical comparison of Average Waiting time and Average Turn Around Time for FCFS, SJF, Priority Scheduling schemas for the input of Burst time (18,6,24,4) with Task4 having the HIGH Priority.

V. CONCLUSION

This paper provides an overview of the basic concepts for extending H.264/AVC towards SVC. Moreover, the basic tools for providing temporal, spatial, and quality scalability are described in detail and experimentally analyzed regarding their efficiency and complexity. We have studied the performance of FCFS, SJF and Priority scheduling aleorithms for dynamic real-time computer system and validated for the best fit mechanism affording user friendliness. We implement and compare the results of three traditional algorithms: FCFS (First Come First Serve), SJF (Shortest Job First) and Priority scheduling algorithm.

ACKNOWLEDGMENT

The authors would like to thank management and Principal, Ghousia College of Engineering, Ramanagaram for their constant support and co-operation in completing this work successfully.

REFERENCES

[1] Mahesh Kumar M.R, Renuka Rajendra. B, Niranjan C.K, Sreenatha .M: "Prediction of Length of the Next CPU Burst in SJF Scheduling Algorithm using Dual Simplex Method", 2nd International Conference on Current Trends in Engineering and Technology, ICCTET'14.

[2] Ms. Tabassum A. Maktum, Ms. Rashmi A. Dhumal, Dr. Lata Ragha: **"**A Genetic Approach for Processor Scheduling", IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014), May 09-11.

[3] Chien-Chi Kao, Shun-Ren Yang, Hsin-Chen Chen : " On Energy Efficiency of IEEE 802.16m Interframe Scheduling for Scalable Video Multicast", IEEE TRANSACTIONS ON MOBILE COMPUTING, DECEMBER 2014

[4] Chao Chen, Robert W. Heath, Alan C. Bovik, Gustavo de Veciana : " A Markov Decision Model for adaptive Scheduling of Stored Scalable Videos", IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, JUNE 2013.

[5] Xiaofei Wang, Min Chen, Ted Taekyoung Kwon, Laurence T. Yang, Victor C.M. Leung : " AMES – Cloud : A framework of Adaptive Mobile Video Streaming and Efficient Social Video Sharing in the Clouds ", IEEE TRANSACTIONS ON CLOUD COMPUTING, 2013.

[6] H. Schwarz and M. Wien, "The Scalable Video Coding Extension of The H. 264/AVC Standard," in *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp.135–141, 2008.
[7] K. Tappayuthpijarn, G. Liebl, T. Stockhammer, and E. Steinbach, "Adaptive Video Streaming over A Mobile Network with TCP-Friendly Rate Control," in *IWCMC*, 2009.
[8] www.streaminglearningcenter.com/
[9] K. Tappayuthpijarn, G. Liebl, T. Stockhammer, and E. Steinbach, "Adaptive Video Streaming over A Mobile Network with TCP-Friendly Rate Control," in *IWCMC*, 2009.

AUTHORS BIOGRAPHY

T VENKATESH, working as Associate Professor in the Department of Computer Science and Engineering, Ghousia College of Engineering, Ramanagaram, since from 1998. His area of interest is Pattern reorganization, Artificial Intelligence, Neural networks, Parallel Computing, Cloud Computing, etc.

SANTHANA G, student of Ghousia College of Engineering, Ramanagaram, is currently studying in 8th semester, Computer Science and Engineering. She is interested in Programming. Her major focus is on doing Research in Cloud technologies.

SWATHI L S, currently studying in 8th semester, Computer Science and Engineering, Ghousia College of Engineering, Ramanagaram, has keen interest in computer networks and web designing.

SUSHMASHREE S, currently studying in 8th semester, Computer Science and Engineering, Ghousia College of Engineering, Ramanagaram, is interested in computer networks and its applications.

ARPITHA C, currently studying in 8th semester, Computer Science and Engineering, Ghousia College of Engineering, Ramanagaram, has keen interest in computer graphics and animations.