An FPTAS for managing playout stalls for multiple video streams in cellular networks

Swapnoneel Roy
School of Computing
University of North Florida
Jacksonville, FL 32224

Anand Seetharam
Computer Science and Information Technology Program,
California State University Monterey Bay
Seaside, CA 93955

***Abstract*—With the proliferation of wireless cellular technology (e.g., 5G, 4G LTE), managing quality of experience (QoE) for wireless clients streaming different videos over these networks is becoming increasingly important. In this paper, we consider the problem of managing playout stalls experienced by multiple clients streaming different videos from a cellular base station. We build on our prior work [1], where we formulate an epoch based lead aware multiple video transmission (LMVT) problem to minimize the total number of playout stalls experienced by mobile clients. In [1], [2], we show that the LMVT problem is NP-hard and propose a pseudo polynomial dynamic programming solution and a simple greedy heuristic to solve the problem. In this paper, we first show that even a simpler and less restrictive version of the LMVT problem remains hard. We then design a fully polynomial time approximation scheme (FPTAS) for the LVMT problem by demonstrating that it is equivalent to the SANTA CLAUS PROBLEM [3], [4]. The FPTAS provides a bounded performance guarantee, differing by a factor of $\frac{1}{1+\varepsilon B}$ from the optimal solution (where B is the number of time slots in an epoch and ε is a small constant).**

I. INTRODUCTION

With the growth of wireless cellular technology, managing quality of experience (QoE) for mobile clients streaming different videos from a cellular base station has become increasingly important (Figure 1). We assume that the base station stores pre-encoded videos and upon receiving requests, streams videos to different clients. Each video consists of a sequence of frames and if a mobile client does not receive a frame by its playout time (the time by which a particular frame has to be received at the client), the client stalls playout until it receives the subsequent frames.

In this paper, we consider a scenario where multiple mobile clients are streaming different variable bit rate (VBR) videos from a base station over a time varying wireless channel. *Our goal is to design an efficient scheduling algorithm so as to minimize the total number of playout stalls across all clients.* We assume that time is divided into slots and scheduling decisions are taken at beginning of an epoch (which consists of multiple slots). We build on our prior work [1], where we formulate this problem as an optimization problem (called the LMVT problem) that takes into account the VBR nature of the video streams and varying channel quality at the clients and allocates slots so as to maximize the minimum playout lead in time across all videos in an epoch [1]. In [1], [2], we showed that the LMVT problem is NP-hard, and proposed a psuedo polynomial dynamic programming solution and an efficient greedy heuristic to solve the problem.

/978-1-4799-7795-6/15/$31.00 ©2015 IEEE

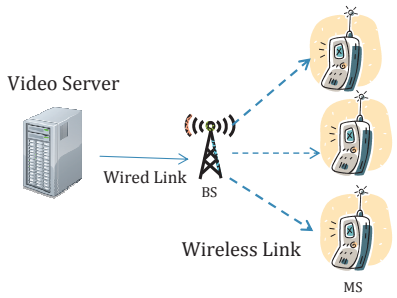

Figure 1. A video streaming system

Our results: Our contributions in this paper are as follows. 1) We first show that a simpler and restrictive version of the LMVT problem is hard. 2) We then design a fully polynomial time approximation scheme (FPTAS) for the LVMT problem by demonstrating that it is equivalent to the SANTA CLAUS PROBLEM [3], [4], [5], [6]. The previous algorithms proposed for the LMVT problem (e.g., the greedy algorithm in [1]) do not have any approximation guarantee. Nor it was known, whether the LMVT problem is hard to approximate, or admits an FPTAS. Our FPTAS provides a bounded performance guarantee, differing by a factor of $\frac{1}{1+\varepsilon B}$ from the optimal solution (where B is the number of time slots in an epoch and ε is a small constant), while the greedy heuristic proposed in [1] does not.

Before, delving into our problem in detail, we briefly describe some of the related work in this space. Scheduling algorithms for improving user QoE in cellular networks have been proposed in literature; in [7], [8], the authors have proposed greedy heuristics for optimizing mean opinion scores for resource allocation in wireless networks. Our work is closely related to the work presented in [9] where the authors determine upper and lower bounds for stall-free video display. But, unlike our work, they consider a single video stream.

II. PROBLEM STATEMENT

In our streaming system (Figure 1), we assume that multiple mobile users are streaming different VBR videos from a base station over a time varying wireless channel. We assume that time is divided into slots and users can receive different bit rates across time slots over the wireless channel. To also accommodate for spatial variation in wireless channel quality, we assume that bit rates for different users during a a particular

time slot could be different. Our goal is to design a scheduling algorithm that executes periodically at the beginning of each epoch (an epoch consists of multiple slots) and assigns time slots to different users in such a way so as to minimize the total number of playout stalls across all users.

Directly addressing the problem of stalls is difficult [1], and so in our optimization framework, the server attempts to distribute the stalls fairly among all clients by maximizing the minimum playout lead in time. The playout lead in time of a video corresponds to the amount of time a client can play out a video without stalling using the data in its buffer, if it does not receive any additional data. Note that because of the VBR nature of the video streams, playout lead in terms of bits does not directly correspond to playout lead in terms of time and thus maximizing playout lead in bits does not lead to fair allocation of playout stalls [1]. In reality the mapping between bits and time is via some non-linear function ϕ.

Let us assume that there are n video streams and let l_i and L_i denote the playout lead in time at the beginning and end of the epoch respectively. Note that l_i is a known constant at the beginning of an epoch. Let V_i denote the number of bits received during an epoch. Therefore, $L_i = l_i + \phi(V_i)$. We assume that an epoch has B time slots and let r_{ij} denote the bit rate received by the i^{th} user in the j^{th} time slot. Let s_{ij} be a binary variable denoting that the i^{th} user is allocated the j^{th} slot. Therefore, we have $V_i = \sum_{j=1}^{B} s_{ij} r_{ij}$. Thus the scheduling problem (LMVT) which executes at the beginning of each epoch can be stated as

$$\begin{aligned} &\textit{Objective: } \max \min\{L_1, \ldots, L_n\} \\ &\text{subject to} \\ &\quad 1.\ s_{ij} \in \{0,1\}\ ,\ \forall i \leq n, \forall j \leq B \end{aligned} \tag{1}$$

III. Background

In this section, we briefly describe some of our prior contributions in [1]. In [1], we showed that the LMVT problem is NP-Hard for a specific function ϕ which maps lead in terms of bits received to the lead in terms of playout time. The decision version of the LMVT problem is stated below.

INPUT: n videos in the channel, B slots in the epoch, the bit rates $r_{ij} \in \mathbb{Z}^+$ for the n videos, a function $\phi(b)$ for calculating the lead for b bits, and $k \in \mathbb{Z}^+$.

QUESTION: Is there an allocation of the B slots to the n videos so that each video has a lead of at least k?

The hardness can be proved by reducing the SUBSET SUM problem to the LMVT problem. A pseudo polynomial time dynamic programming solution for the problem is designed in [2]. The exponential complexity of the dynamic programming solution makes it infeasible to implement in practice. We then propose a simple playout lead-aware greedy heuristic to solve the problem and demonstrate practical efficacy of the greedy algorithm via trace based evaluation [1], [2]. We present the dynamic programming solution for this problem below and then proceed to highlight our contributions in this work.

Input: n, the number of videos, B, the number of slots, r_{ij}, the rate of video v_i for slot j
Output: An allocation of the B slots over n videos where the minimum number of bits received by a video is maximized

Generate the $\prod_{i=1}^{n}({b_i}^{max}+1)$ vectors where ${b_i}^{max} = \sum_{j=1}^{B} r_{ij}$;
Construct the $\prod_{i=1}^{n}({b_i}^{max}+1)$ by n matrix of the vectors;
Have the truth value vector of length $\prod_{i=1}^{n}({b_i}^{max}+1)$;
Initialize the truth value of $< 0, \cdots, 0 >$ to $true$ and the rest to $false$;
for m = 1 *to* B **do**
  foreach Tx *vector* T **do**
    if $F(m-1, T)$ **then**
      Set $F(m, T)$ to $true$;
    end
    else
      for $i = 1$ *to* n **do**
        if $F(m-1, W_i)$ **then**
          Set $F(m, T)$ to $true$;
          break;
        end
      end
    end
  end
end
Return the Tx vector T with the maximum $min(b_i)$ value and with $F(B, T)$ true;

Algorithm 1: The exact dynamic programming algorithm

The Exact Dynamic Programming Algorithm

Define ${b_i}^{max}$ to be the maximum number of bits that video v_i can receive in an epoch. In other words, ${b_i}^{max} = \sum_{j=1}^{B} r_{ij}$, is the number of bits v_i would receive, if *all* the B slots are allocated to it. Given m slots and n videos, a Tx (transmission) vector is an *n-tuple* $< b_1, \cdots, b_n >$ which tells us whether a slot allocation is possible such that video v_i receives at least b_i bits in the allocation. The length of the Tx vector is the number of videos n. We define the predicate $F(m, T)$ for m slots and Tx vector T. $F(m, T)$ is true iff an allocation can achieve Tx. For two Tx vectors T_1, and T_2, we define $T_1 \preceq T_2$ iff $T_1[i] \leq T_2[i]$, $\forall i$. It is easy to see, if $F(m, T_2)$, then $F(m, T_1)$. In the dynamic programming, we generate $\prod_{i=1}^{n}({b_i}^{max}+1)$ possible Tx vectors starting from $< 0, \cdots, 0 >$ till $< {b_1}^{max}, \cdots, {b_n}^{max} >$. For each video v_i, we have the values taken from the set $\{0, 1, \cdots, {b_i}^{max}-1, {b_i}^{max}\}$. We maintain an $\prod_{i=1}^{n}({b_i}^{max}+1)$ by n matrix of the vectors during the execution of the

dynamic programming algorithm. Also, we have a *truth value* vector of length $\prod_{i=1}^{n}({b_i}^{max}+1)$. Each cell in the true value vector corresponds to the value of $F(m,T)$ for m slots, and Tx vector T. We initialize the truth value of $<0,\cdots,0>$ to $true$ and the rest to $false$. This signifies that we can always achieve vector $<0,\cdots,0>$, even without any slot allocation. We then start from $m=1$ to the total number of slots B, and evaluate the truth values. The truth values ($F(B,T)$) at the end tell us whether that vector T was achievable by a slot allocation with the B slots. We then choose the vector with the maximum minimum b_i value as our solution, and have the corresponding slot allocation as the optimal answer. The way to evaluate $F(m,T)$ is as follows:

1) If $F(m-1,T)$, then $F(m,T)$.
2) Else let W_i be the vector where all the positions of W_i except $W_i[i]$ is equal to T. $W_i[i]=max(0,T[i]-r_{im})$. For $i=1$ to n, if $F(m-1,W_i)$, then $F(m,T)$.

As a final step, we store the s_{im} value for each $1\leq m\leq B$ (and $1\leq i\leq n$). We maintain an allocation pointer to a Tx-vector T for which $F(m,T)$ is true at the end of step m, which tells us which video was the current slot allocated to. At the end, the allocation pointers give us the allocation which leads to the optimal solution. Specifically, allocation pointers backtracked from the optimal solution vector to initial vector $<0,\cdots,0>$ give us an optimal slot allocation.

We present the algorithm in Algorithm 1. Algorithm 1 has a runtime of $O(Bn\prod_{i=1}^{n}({b_i}^{max}+1))$. Since $\prod_{i=1}^{n}({b_i}^{max}+1)$ can be exponentially large, Algorithm 1 has an exponential runtime.

IV. Hardness of LMVT problem for linear ϕ

In this work, we have two major contributions. 1) We first show the LMVT problem to remain NP-Hard even for a simpler scenario when ϕ is linear. 2) We then design an FPTAS for the LMVT problem which provides strict performance guarantees.

We assume a linear function ϕ to calculate the lead based on the number of bits received, b. Let us consider a monotonic linear ϕ, such that $\phi(b)=b$. The problem then simplifies to finding a slot allocation to maximize the minimum number of bits received by any video. We show that the problem remains NP-Hard, even in this scenario.

We reduce the Partition Problem, a known NP-Hard problem to LMVT. The Partition Problem can be presented as:

> Partition Problem
> Input: A set $S\subseteq\mathbb{Z}^+$ with $\sum_{x\in S}x=U$.
> Question: Can S be partitioned to S' and $S\setminus S'$ such that $\sum_{x\in S'}x=\sum_{x\in S\setminus S'}x=U/2$?

For the reduction, consider any instance of the Partition Problem with $S=\{x_1,x_2,\cdots,x_B\}$, $\sum_{x\in S}x=U$, and $|S|=B$. Now consider an instance of LMVT where we have 2 videos v_1 and v_2, B slots, and the bit rates $r_{1j}=r_{2j}=x_j\in S$, for each slot j. Note that $|S|=B=\#$ *of slots for the* LMVT *instance*. Set $k=U/2$.

Lemma IV.1. *The above instance of the* Partition Problem *has a solution iff the instance of* LMVT *has a solution.*

Proof: We show that the instance of the Partition Problem has a solution iff we can find a slot allocation for the 2 videos of the LMVT instance, such that the number of bits received by each video is exactly $U/2$.

Suppose the Partition Problem has a solution. That is, we have S' and $S\setminus S'$ such that $\sum_{x\in S'}x=\sum_{x\in S\setminus S'}x=U/2$. We find a slot allocation of the B slots which allocates slot i to video v_1 if $x_i\in S'$. Otherwise i is allocated to v_2. We note that all the B slots get allocated this way, since $|S|=B$. Now since $\sum_{x\in S'}x=\sum_{x\in S\setminus S'}x=U/2$, it is easy to see that the number of bits received by v_1 and v_2 is exactly equal to $U/2$.

For the other way, suppose we have a slot allocation such that number of bits received v_1 and $v_2=U/2$. We partition S in the following way:

1) If slot i is allocated to v_1, then $x_i\in S'$.
2) Else $x_i\in S'\setminus S$.

Since $\sum_i r_{1i}=\sum_i r_{2i}=U/2$, we have $\sum_{x\in S'}x=\sum_{x\in S\setminus S'}x=U/2$, and hence a solution to the instance of Partition Problem. □

V. An FPTAS for LMVT

In section III, we presented the pseudo polynomial dynamic programming solution for the LMVT problem. The exponential complexity of the solution, prompts us to design a FPTAS with performance guarantees. Therefore, in the FPTAS, instead of considering all the values in the set $\{0,1,\cdots,{b_i}^{max}-1,{b_i}^{max}\}$ for each video v_i, we discretize the set to the powers of $1+\varepsilon$, where $\varepsilon>0$. We define the function $\psi(i)=\lfloor(1+\varepsilon)^{\lfloor log_{1+\varepsilon}i\rfloor}\rfloor$. Now we have the set for each video v_i, as $\{0,\psi(1),\cdots,\psi({b_i}^{max}-1),\psi({b_i}^{max})\}$. Clearly, we have at most $log_{1+\varepsilon}({b_i}^{max}+1)$ values in the set. Hence, we have only $\prod_{i=1}^{n}log_{1+\varepsilon}({b_i}^{max}+1)$ Tx vectors to evaluate truth values for, in the FPTAS. In the evaluation of $F(m,T)$, instead of considering W_i as in Algorithm 1, we consider W_i' where W_i' is the vector where all the positions of W_i' except $W_i'[i]$ is equal to T. $W_i'[i]=max(0,\psi(T[i]-r_{im}))$. We present the FPTAS in Algorithm 2.

Algorithm 2 considers only $\prod_{i=1}^{n}log_{1+\varepsilon}({b_i}^{max}+1)$ to evaluate out of the $\prod_{i=1}^{n}({b_i}^{max}+1)$ Tx vectors evaluated by Algorithm 1.

Lemma V.1. *The error generated by the rounding of W_i in Algorithm 1 to W_i' in Algorithm 2 is at most $\frac{1}{1+\gamma}$ where $\gamma>0$.*

Proof: Suppose we have $<b_1,\cdots,b_n>\iff<c_1,\cdots,c_n>$ from the $F(m,T)$ evaluation step of Algorithm 1. In other words $F(m,T_1)$ for $T_1=<c_1,\cdots,c_n>$ has been evaluated to $true$ because $F(m-1,T_2)$ had been evaluated to be $true$ for $T_2=<b_1,\cdots,b_n>$. Hence,

Input: n, the number of videos, B, the number of slots, r_{ij}, the rate of video v_i for slot j
Output: An allocation of the B slots over n videos where the minimum number of bits received by a video is maximized

Generate the $\prod_{i=1}^{n} log_{1+\varepsilon}({b_i}^{max}+1)$ vectors where ${b_i}^{max} = \sum_{j=1}^{B} r_{ij}$;

Construct the $\prod_{i=1}^{n} log_{1+\varepsilon}({b_i}^{max}+1)$ by n matrix of the vectors;

Have the truth value vector of length $\prod_{i=1}^{n} log_{1+\varepsilon}({b_i}^{max}+1)$;

Initialize the truth value of $< 0, \cdots, 0 >$ to $true$ and the rest to $false$;

for m = 1 *to* B **do**
  foreach Tx *vector* T **do**
    if $F(m-1, T)$ **then**
      Set $F(m, T)$ to $true$;
    end
    else
      for $i = 1$ *to* n **do**
        if $F(m-1, W_i')$ **then**
          Set $F(m, T)$ to $true$;
          break;
        end
      end
    end
  end
end

Return the Tx vector T with the maximum $min(b_i)$ value and with $F(B, T)$ true;

Algorithm 2: An FPTAS for LMVT PROBLEM

$\exists i \in [n]$, such that, $T_2[i] = b_i = max(0, c_i - r_{im})$, and for all other positions j, we have $T_2[j] = T_1[j]$.

Now suppose we have $T_2' = < b_1', \cdots, b_n' >$ and $T_1' = < c_1', \cdots, c_n' >$ in the table for Algorithm 2, where $b_i' = \psi(b_i)$, and $c_i' = \psi(c_i)$, $\forall i \in [n]$. We want to show that if Algorithm 2 evaluates $F(m, T_1')$ to $true$ if $F(m-1, T_2')$ was evaluated to $true$ at an earlier step, with an error of at most $\frac{1}{1+\gamma}$. In other words, $< b_1', \cdots, b_n' > \iff < c_1', \cdots, c_n' >$ in Algorithm 2 with an error of at most $\frac{1}{1+\gamma}$.

We observe that $T_2'[j] = T_1'[j]$ for all $j \in [n] \setminus \{i\}$. For $j = i$ we have $T_2'[i] = b_i' = \psi(b_i) = max(0, \psi(c_i - r_{im})) \geq \psi(c_i - r_{im})$. Algorithm 2 would calculate the value of position i for vector W_i' as $W_i'[i] = max(0, \psi(c_i' - r_{im})) \geq \psi(c_i' - r_{im})$. We have $\psi(c_i' - r_{im}) = \psi(\psi(c_i) - r_{im}) \geq \psi(\frac{c_i - r_{im}}{1+\varepsilon}) \geq \frac{1}{1+\gamma}\psi(c_i - r_{im})$, where $\gamma = 2\varepsilon$. □

Lemma V.2. *Algorithm 2 has a runtime of* $O(Bn\prod_{i=1}^{n} log_{1+\varepsilon}({b_i}^{max}+1))$.

Lemma V.3. *The value of the solution* (S_{fptas}) *returned by Algorithm 2 differs from that* (S_{opt}) *returned by Algorithm 1 at most by a factor of* $\frac{1}{1+\varepsilon B}$.

Proof: At any step i, the value of any position of any vector of Algorithm 2 differs from the corresponding position of the corresponding vector for Algorithm 1 by a factor of $\frac{1}{(1+\varepsilon)^i} \approx \frac{1}{1+\varepsilon i}$ due to the rounding. We perform this rounding B times. Hence the values of the vectors after the full execution would differ by a factor of at most $\frac{1}{1+\varepsilon B}$. Lemma V.2 and V.3 lead to the following theorem. □

Since we look at $O(Bn\prod_{i=1}^{n} log_{1+\varepsilon}({b_i}^{max}+1))$ values, the maximum storage space needed (for the allocation pointers) to implement Algorithm 2 is polynomially bounded.

Theorem V.4. *Algorithm 2 is an FPTAS for* LMVT PROBLEM.

VI. CONCLUSION AND FUTURE WORK

In this paper, we studied the problem of minimizing the number of playout stalls experienced by clients streaming different videos from a cellular base station. We built on our prior work, where we formulated an epoch based lead aware optimization framework (LMVT) which maximized the minimum playout lead of the mobile clients to improve the overall quality of experience. In this paper, we demonstrated that even a restrictive and simplified version of the LMVT problem is NP-hard and proposed an efficient fully polynomial time approximation scheme (FPTAS) for the LMVT problem. In future, we plan to perform a realistic trace based evaluation (with real wireless network and real video traces) to compare the performance of the proposed FPTAS with the greedy heuristic and the pseudo polynomial optimal solution [2].

REFERENCES

[1] P. Dutta, A. Seetharam, V. Arya, M. Chetlur, S. Kalyanaraman, and J. Kurose, "On managing quality of experience of multiple video streams in wireless networks," in *IEEE INFOCOM 2012*.

[2] P. Dutta, A. Seetharam, V. Arya, J. Kurose, M. Chetlur, and S. Kalyanaraman, "On managing quality of experience of multiple video streams in wireless networks," *To appear in IEEE Trans. on Mobile Computing*.

[3] N. Bansal and M. Sviridenko, "The santa claus problem," in *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, 2006, pp. 31–40.

[4] M. Bateni, M. Charikar, and V. Guruswami, "Maxmin allocation via degree lower-bounded arborescences," in *Proceedings of the 41st annual ACM symposium on Theory of computing*, 2009, pp. 543–552.

[5] D. Chakrabarty, J. Chuzhoy, and S. Khanna, "On allocating goods to maximize fairness," in *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, 2009, pp. 107–116.

[6] B. Haeupler, B. Saha, and A. Srinivasan, "New constructive aspects of the lovasz local lemma," *Journal of the ACM (JACM)*, vol. 58, no. 6, p. 28, 2011.

[7] M. Shehada, S. Thakolsri, Z. Despotovic, and W. Kellerer, "Qoe-based cross-layer optimization for video delivery in long term evolution mobile networks," in *WPMC*, 2011.

[8] S. Thakolsri, S. Khan, E. Steinbach, and W. Kellerer, "Qoe-driven cross-layer optimization for high speed downlink packet access," *Journal of Communications*, vol. 4, no. 9, 2009.

[9] G. Liang and B. Liang, "Effect of delay and buffering on jitter-free streaming over random VBR channels," *IEEE Trans. on Multimedia*, vol. 10, no. 6, 2008.