Signal Processing: *Image Communication* 27 (2012) 513–521

Contents lists available at SciVerse ScienceDirect

Signal Processing: *Image Communication*

journal homepage: www.elsevier.com/locate/image

A game theoretic approach to minimum-delay scalable video transmission over P2P

Stefano Asioli *, Naeem Ramzan, Ebroul Izquierdo

Multimedia and Vision Research Group, Queen Mary, University of London, Mile End Road, E1 4NS London, UK

ARTICLE INFO

Available online 19 March 2012

Keywords:
Peer-to-peer
Scalable video coding
Game theory
Delay minimization

ABSTRACT

In this paper we describe a game theoretic framework for scalable video streaming over a peer-to-peer network. The proposed system integrates minimum delay functionalities with an incentive provision mechanism for optimal resource allocation. First of all, we introduce an algorithm for packet scheduling that allows users to download a specific sub-set of the original scalable bit-stream, depending on the current network conditions. Furthermore, we present an algorithm that aims both at identifying free-riders and minimising the transmission delay. Uncooperative peers are cut out of this system, while users upload more data to those which have less to share, in order to fully exploit the resources of all peers. Experimental evaluation shows that the proposed model can effectively cope with free-riders and minimise the transmission delay for scalable video transmission by exploiting a packet scheduling algorithm, game theory, and a minimum-delay algorithm.

1. Introduction

In the past few years, popularity of multimedia applications over the Internet has grown exponentially. This is due to the increasing diffusion of broadband connections. Highly demanding services for video-on-demand or live streaming are now offered by several providers and most of them usually rely on a client/server architecture. However, this approach is expensive and it does not exploit the idle resources of the users of the system. On the other hand, peer-to-peer (P2P) is a valid alternative to this model. In fact, if peers are given the proper incentives, they can share their own resources, getting a reward from other users and lowering the burden of the server.

Live video streaming over P2P networks, however, still presents some challenges. First of all, data chunks have strict deadlines, represented by their playback time. That is, in real-time applications like P2P TV broadcasting, the delay between the generation of a chunk and the receiving time should be minimised. Second, users have different upload capacities and P2P networks are highly heterogeneous. Finally, users in this system can behave as free-riders. In fact, knowing the vulnerabilities of a P2P protocol, they download data without returning anything in exchange.

The problem of network heterogeneity can be partially solved by using scalable video coding (SVC). This allows to adapt the content to different users requirements by selecting an appropriate sub-set of the original sequence for download and discarding the rest. These codecs are based on the DCT transform, like the standard H.264/SVC [1], or the wavelet transform, like our wavelet-based SVC [2]. In both cases, adaptation can be performed in terms of frame size, frame rate or Signal-to-Noise Ratio (SNR).

P2P systems for video streaming can be divided into two categories: tree-based and mesh-based. Many algorithms for minimum-delay streaming [3,4] rely on a tree-based architecture. This approach has several advantages, including better resource allocation. However, especially

* Corresponding author.
E-mail addresses: stefano.asioli@eecs.qmul.ac.uk (S. Asioli), naeem.ramzan@eecs.qmul.ac.uk (N. Ramzan), ebroul.izquierdo@eecs.qmul.ac.uk (E. Izquierdo).

0923-5965/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.image.2012.02.012

in big networks, these trees might be computationally expensive to maintain. Moreover, these systems only have limited flexibility. The most critical aspect is, however, the lack of resource reciprocation between a child and a parent node, as communication usually flows only in one direction. A solution to this problem is represented by complementary trees [3]. Nevertheless, this system not only needs to face the usual maintenance costs, but also needs to create several overlay networks that need to be constantly balanced. On the other hand, there exist solutions based on a mesh topology [5–7]. Specifically, [5] is a *push-based* solution where the latest useful chunks are forwarded to the *most deprived* peers. Experimental evaluation [5] shows that this technique can achieve results that are comparable to tree-based approaches.

The problem of resource reciprocation in P2P networks has been an important area of research for many years. Several models, like the original BitTorrent (BT) [8] focus on short-term effects, which might not be suitable for video related applications. In fact, peers might not always have data to share with users they are currently downloading from. Moreover, studies like [9] show that BT is vulnerable to the problem of free-riding. Many alternative approaches are based on game theory [10], like [11–13]. For example, in [11], Park et al. consider the problem of content production and sharing together. They also analyse under which conditions it is convenient for a peer to cooperate in a P2P system and a few pricing schemes are also proposed. In addition, in [12] a credit-line mechanism is introduced. In this solution, peers always cooperate with other users, unless the difference between the amount of data they have uploaded to another peer and the amount of data they downloaded from it exceeds a given threshold. It is possible to prove that this strategy is cheat-proof, a Nash Equilibrium and strongly Pareto optimal [10]. Finally, in [13] a foresighted resource reciprocation model is illustrated. It is based on the idea that peers should upload data to other users considering how this action will impact their behaviour in the future. Moreover, peers do not just decide which users they want to cooperate with, but also the scale of this contribution. This is achieved using a Markov Decision Process [14] framework.

The proposed system is a game theoretic framework for scalable P2P video streaming with a focus on optimal resource allocation and delay minimisation. Our idea is to integrate minimum-delay streaming functionalities within an incentive provision mechanism. In addition to this, our system is designed for the transmission of scalable video sequences. Scalable video coding forms an integral part of our framework, as peers are discouraged from downloading videos with a quality which is higher than the one they can provide other users with. To the best of our knowledge, optimising streaming rates and finding incentives for users to cooperate are usually considered as separate problems. Most real systems, however, need to deal with both at the same time. Our technique considers both issues, while aiming at achieving nearly optimal performance.

The remaining Sections of this paper are organised as follows: Section 2 provides some background; Section 3 describes the proposed algorithm; Section 4 presents the results obtained in our simulations; Finally, Section 5 concludes the paper.

2. Background

The problem of finding incentives in P2P networks has been extensively studied in the past few years [9]. More recently, some systems have been specifically designed for video streaming. A common factor of all these solutions is that each peer is associated with a *utility* or *payoff function* [15]. Let us suppose that a video bit-stream of bit-rate q is divided into chunks and this video is originally stored in a server; for simplicity, time is divided into rounds. First of all, the two-player game will be analysed, where the utility function for a single round is defined as follows [15]:

$$u_1(x_1,x_2) = x_2\gamma_1 - x_1\lambda_1$$

$$u_2(x_1,x_2) = x_1\gamma_2 - x_2\lambda_2 \quad (1)$$

Eq. (1) contain several terms. First of all, x_i is the action taken by peer P_i and it is either 1 or 0, depending on the decision of i to cooperate or defect. $\gamma \in [0,1]$ denotes the gain which a certain user obtains from the receiving a video chunk and it is a subjective quantity. On the other end, λ_i is the cost (loss) associated to cooperation and it is can be seen in terms of bandwidth used to transmit one chunk in one round. There are two terms in the equation. The first one is the gain of user P_i with respect to the action of the other peer, while the second is the cost, which depends on their own action. If the game is only played for one round, the only action profile (x_1, x_2) that satisfies the Nash Equilibrium condition is (0,0) [12]. The same holds if the game is played for a finite number of times and the termination time is known to both users. However, in these systems users do not exactly know when the other peers will leave the game. Therefore, other Nash Equilibria exist and the game can be modelled as an infinitely repeated game.

In this case [12] an averaged utility function $U_i(\mathbf{x})$ is considered, where $\mathbf{x} = (\mathbf{x}_1,\mathbf{x}_2)$ is a vector containing the decisions of the peers at each round. This function is defined as

$$U_i(\mathbf{x}) = \frac{\sum_{t=1}^{T_{fin}} u_i(\mathbf{x}_1(t),\mathbf{x}_2(t))}{T_{fin}} \quad (2)$$

where $\mathbf{x}_i(t)$ is the action of peer P_i at round t and T_{fin} can be either the round corresponding to the end of the playback or the round in which peers stop cooperating. If both peers follow random strategies until the end of the video, the possible outcomes of the repeated game are all the points $(u_1,u_2) \in V_U$ inside the convex quadrilateral shown in Fig. 1. However, in [12] each peer stops cooperating with the other user when the other peer refused to cooperate in the previous round or if the other user cannot share any data that can be useful for the current peer. This will also become the final round T_{fin} used to calculate the final utility profile. If both peers cooperate until the end of the game, the corresponding utility function will be $U(\mathbf{x}) = (U_1(\mathbf{x}),U_2(\mathbf{x})) = (\gamma_1-\lambda_1,\gamma_2-\lambda_2) = \mathbf{u}$, otherwise, it will