Multimedia Systems manuscript No.
(will be inserted by the editor)

A Bio-inspired Managed Video Delivery Service using HTTP-based Adaptive Streaming

Yusuf Sani · Jason J. Quinlan · Cormac J. Sreenan

Received: date / Accepted: date

Abstract As consumers switch to video-on-demand services, over the best effort Internet, the importance of service level agreement enforcement schemes can not be over emphasised. For these agreements to be effective, content providers must be able to enforce business policies in a simple and scalable manner, typically without access to the functionality within the core of the content delivery infrastructure. The option of relying on Media Presentation Description (MPD) attributes for video rate restriction is neither flexible or effective. Hence, in this paper, we present a bio-inspired solution that exploits the inherent features of an HTTP-based adaptive streaming service to enable content providers guarantee service level agreements. We utilise concepts from mathematical ecology that model species competing for a limited resource. In the proposed solution, distributed clients are assisted with global information using SDN. To enhance the scalability of the system, a business policy is enforced through parameter optimisation. To demonstrate the applicability of the proposed service, we built a test-bed and

Yusuf Sani
University College Cork
School of Computer Science & Information Technology
Western Rd
Cork
Ireland
E-mail: ys8@cs.ucc.ie

Jason J. Quinlan
University College Cork
School of Computer Science & Information Technology
Western Rd
Cork
Ireland
E-mail: j.quinlan@cs.ucc.ie

Cormac J. Sreenan
University College Cork
School of Computer Science & Information Technology
Western Rd
Cork
Ireland
E-mail: cjs@cs.ucc.ie

implemented a number of business policies. Evaluation results show that business policies are enforced in a fair and stable manner.

Keywords Population Dynamics · HTTP Adaptive Streaming · Dynamic Adaptive Streaming over HTTP · SDN

1 Introduction

To cater for heterogeneous viewing devices and unpredictable network conditions, HTTP-based adaptive streaming (HAS) services, such as the Dynamic Adaptive Streaming over HTTP (DASH) [23], are the most popular mechanisms for video distribution over the Internet. HAS services require content providers to segment their video file into small chunks, with each chunk encoded into multiple bit-rates. This allows a HAS client to progressively choose a chunk with a video rate that best suits its context.

Since HAS services are generally delivered via the best-effort Internet by the content providers, either directly or through content delivery networks, guaranteeing an agreed quality level is difficult. Providing a service level guarantee is further complicated by the fact that, in practice, video content providers typically adopt a multi-tiered service strategy, where members of a group of customers are individually and separately charged a particular price in return for an agreed level of service.

A video quality guarantee can be enforced by a central network control agent or through a client-side mechanism. Amongst the network-based solutions is the Server and Network Assisted DASH (SAND) [17], which presents a standardised interface for message exchange between streaming clients and network elements. Like most of the existing network-assisted designs, SAND mandates an active network agent, a DASH aware network element (DANE), but this design paradigm implicitly assumes that a content provider has access to the core of the content delivery network. For example, implementing a shared resource allocation service using SAND the authors of [18] assume the service intended deployment scenario is either a home wireless network, or aircraft. However, at the client-side, a content provider can provide differentiated video quality levels to its customers by either restricting the maximum advertised bitrate in the media description document (MPD) sent to a client, or by using the PROFILES attribute provided in ISO/IEC 23009, which "permits external organisations or individuals to define restrictions" on the features that a client can consume [1]. As we shall see later in Section 7, these techniques fail when the available capacity is not sufficient to allow all clients to converge at the MPD advertised upper limit.

Hence, we propose to define a scalable business policy enforcement model without the need for an active network agent, thus removing the necessity of access to functionality within the network core. We envision a system where individual streaming clients rely on local information about their environment to assist in making decisions that converge at a service level agreement (SLA) determined rate. To achieve this, we take inspiration from nature. In a typical biological system, the behaviour of the "global dynamics" emerges from simple interactions amongst local individuals, which initially may be in a state of disorder [3]. A typical natural scenario, where self-organisation is evident in nature, is population dynamics. Here, out of local actions of various species, global order and regularity are borne. And under well-known conditions, the ordered global state is known to be in a stable equilibrium.

As such, we utilise concepts and models from population dynamics to design a coordinated distributed control framework for enforcing a differentiated treatment to different subscribers of a streaming service. Specifically, we use the Lotka-Volterra (LV) competition model [33]. A set of differential equations used in modelling the impact of competition, against a single limited resource, on the population dynamics of the competing species.

The proposed scheme accords the same treatment to clients within a given class and offers different treatment to clients belonging to different classes. In this paper, we define a class to be a set of streaming clients, whose users sign and agree to a similar service level agreement, such that flows of each individual class are accorded similar treatment by a video content provider, for example, are allocated equal network resource. The proposed scheme combines a client-side video rate selection utility function with information on the global/cumulative transmission rate of the competing flows at the bottleneck link. The framework allows HAS video clients, sharing a network, to focus on short-term decisions such as the next video rate to request, while allowing a passive central control agent to guide the clients on longer horizon decisions e.g. the target video rate at which to converge.

The rest of the paper is arranged as follows. Section 2 presents related work. Section 3 introduces population dynamics, and specifically the Lotka-Volterra competition model. While in Section 4 we propose the system model and how we map our system to its ecological counterpart. In Section 5 the behaviour of the system is analytically investigated, using a number of business policies. Section 5.2 discusses how the various parameters of the proposed scheme can be derived. While experimental set-up is discussed in Section 6, and the results of our experimentation are presented in Section 7. Section 8 concludes the paper.

2 Related Work

HTTP-based adaptive video streaming services adapt the quality of the delivered video content to the context of a streaming client. In its standard form, a video file is segmented into equal temporal sized chunks and encoded into multiple bit rates. Details of the chunks (such as URL, bitrate, resolution, etc.) are gathered into an XML-based meta-data file called a media presentation description (MPD) file and are stored in a server. For client chunk selection, an adaptation module of HAS, also known as *Adaptive Bit-rate Selection* (ABR) [21], continuously monitors and measures system capabilities, typically, network capacity and buffer occupancy and then progressively request chucks with an appropriate bitrate.

A plethora of client-side ABR algorithms exist. In [14], a *throughput-based* HAS-based services that strictly rely on throughput estimate for video rate selection is proposed. However, it is known that throughput-based ABRs suffer from sub-optimal video quality, excessive instability and a high number of rebuffering events [8]. An alternative approach solely relies on buffer state changes for video bitrate selection decision [9, 20, 24]. In [9], a piece-wise defined function is used to map video rate to buffer level while in [20] the logistic function is used. The authors of [24] used Lyapunov optimisation to map the bit-rates to the buffer occupancy. To get the best of both buffer-based and throughput-based techniques, hybrid video rate selection algorithms are the most widely deployed [12]. In [31], a control theory based player was proposed, the service relies on both throughput estimation and buffer state changes for video rate selection decision. However, in [15] a deep reinforcement leaning based player was proposed. The scheme uses a model-free trained to generate video quality adaptation algorithms. However, these algorithms, like most client-side ABR schemes, are designed to opportunistically maximise their user-level quality of experience (QoE). Hence, making it difficult to enforce even the most basic level of guarantee: fairness amongst flows. Exceptions to this rule are the client-side schemes such as [12,20] that are designed to prevent clients from selfishly maximising user level QoE, however, they too, are not designed to provide a flexible SLA enforcement.

To guarantee fairness, network-assisted schemes have been proposed. In [6], a software-defined network (SDN) based agent is used to orchestrate the monitoring of video streams and the arbitration of resource allocation to ensure network-wide QoE fairness. The authors of [10] conducted a large scale evaluation of the DASH Assisting Network Element (DANE). Their resulting DANE improves the performance of the streaming clients competing for a limited bandwidth, by reducing stalling and video quality instability. Furthermore, they found the scheme improves the average video bitrate, and reducing unfairness. In [32], a network traffic pacing agent that relies on both network and client state information for traffic management is proposed. While in [19], the use of traffic shaping techniques is found to have a positive impact on the performance of HAS streaming clients. In [4], a network-assisted service is proposed: the scheme leverages the capabilities of the SDN control plane to collect various information, such as available bandwidth, to help guide a player to make a more informed bitrate selection decision. However, all these services are designed to enforce a single policy: equal allocation of resource, using an active network agent.

Some studies have employed bio-inspired techniques to control the rate of traffic injection into a network. In [22], a bio-inspired buffer-based video rate adaptation scheme based on Verhulst population model is proposed. However, the work neither considers the impact of competition when making the video rate selection decision nor guarantee service delivery. In [2], the LV model was used in designing a congestion control algorithm for a wireless sensor network-based application. In [30], multi-priority data transmission control for the Enhanced Distributed Channel Access (EDCA) protocol is proposed. However, these techniques are only designed for a specific scenario. The scheme proposed in [2] treats all traffic equally, while the scheme in [30] assumes traffic differentiation is in strict compliance with the IEEE 802.11e EDCA protocol. Furthermore, both approaches implicitly assumed that a content provider has access to the intermediary network nodes. In this paper, we exploit the inherent nature of the LV model that allows agents acting individually to converge at a global optimal solution. By appropriately tuning the behavioural parameters of our LV model, a converged state can be obtained. Hence, we present a flexible business policy enforcement for HTTP-based adaptive video streaming services. The scheme allows a content provider to scalably implement a variety of business policies without the need for an active network agent.

3 Mathematical Ecology: Population Dynamics

Population dynamics is the branch of mathematical ecology that studies the changes in population as well as the processes that influence the changes, e.g. the availability of resources, the presence of a competing species and the nature of the competition [26]. Modelling population change is commonly initiated with the following basic assumptions:

1. The size of population, at a give time, determines the state of a system.
2. Habitat has a limited caring capacity.
3. There is always a seed population to kick-start the growth.
4. Growth is density-dependant.
5. Impact of exogenous factors, such as competing species and available resource, are consequential.

The most prominent model used in formulating population dynamics is Verhulst model [28], which formulates the dynamics of population growth in absence of competition. However, since rarely does a species of animal live in isolation, this model does have a limited applicability. Furthermore, in a typical ecological habitat, animals are in constant competition with one another for a limited resource. This competition can be within animals of the same species, called *intra-species competition*, or between animals of different species, called *inter-species competition*. The Lotka-Volterra (LV) model [33] extends the Verhulst model and describes the dynamics of population growth when multiple species compete for some limiting resource, such as food, nutrients, space etc.

The LV model relies on three sets of parameters: intrinsic growth rate, which governs the behaviour of a system when only a single species exist. An *intra-species* coefficient that rules the behaviour of intra-species competition; typically members of the same species share equal value. And finally, an *inter-species* coefficient, which is concerned with interactions amongst different species. The n-species LV model is expressed by the following differential equation

$$\frac{dp_i}{dt} = p_i\left(r_i - \frac{\beta_i r_i}{K_i} - \frac{r_i}{K_i}\left(\sum_{j=1}^{n} \alpha_{ij} p_j\right)\right), \tag{1}$$

$$i = 1, ..., n, p_i, K_i, r_i, \beta_i, \alpha_{ij} \in \mathbb{R}.$$

Where p_i is the size of the population of species i at the time t , K_i the maximum population size of i^{th} species that the habitat can sustain and r_i is the intrinsic growth rate constant. β_i is the intra-species competition coefficient and α_{ij} is the inter-species competition coefficient of species j on the growth rate of species i.

4 System Modelling

Consider a case where a set of streaming devices $\mathcal{I} = \{1, ..., i\}$ share a network slice with a total capacity C. Since a network slice is rented out on a fixed capacity basis, the upper-bound of C is known (for the time-dependent estimate of C see Section 6 for more detail). Let each device be characterised by two parameters $\{0 < R_{min} < R_{max}\}$, where R_{min}, R_{max} are the minimum and the maximum bitrate that the resolution of a device permits.

Table 1: Model Notation

p_i	Size of the population of species i
K_i	Maximum population size of i^{th} species
r_i	Intrinsic growth rate constant
α	Inter-species competition coefficient of species
β_i	Intra-species competition coefficient
R_{max}	Maximum video rate
R_{min}	Minimum video rate
B_{max}	Maximum buffer size
B_t	Buffer size at time t
C	Total link capacity
Q	The set of the available video representations
t^s	Chunk start time
t^e	Chunk end time
$x_i(t)$	Download rate of client i at time t
X	$diag(x_1, x_2, x_3, ..., x_j)$
R	$(r_1, r_2, r_3, ..., r_i)^T$
A	see Eq. 5
$w_i(t)$	see Eq. 7
x^*	set of positive non-zero stable equilibrium

When a client begins streaming using HAS technology, typically at time $t = 0$, a server presents an MPD file to the streaming client. This file contains information about the set of available video rates $Q = \{q_0, q_1, q_2, \ldots, q_n\}$. Let us assume that the bitrates are arranged in increasing order, such that $q_0 = R_{min}$ and $q_n \leqslant R_{max}$. In other words, a client can only choose from a subset of available video rates. Let B_t be the buffer occupancy at time t and B_{max} be the maximum buffer size. After parsing the MPD file at time $t_s > t_0$, a client begins to request chunk $l \geqslant 1$ with the video rate $q_n < R_{max}$. The chunk download starts at time t^s and finishes time t^e. This continues until either the buffer is full or the last chunk is requested. Let $x_i(t)$ be the download rate of client i at time t. When $x_i(t) < R_{max}$ $\forall i$ and $\sum_{i=1}^{i} x_i(t) < C(t)$ resource allocation is easy, as all devices can stream at R_{max}. However, if total bandwidth is not sufficient to cover such a scenario, in a managed service, a provider will want to ensure that bandwidth is allocated such that system converges at $x_i(t) \in [R_{min}, R_{max}]$ $\forall i$ in accordance with a business policy (see Table 1 for the summary of notations). For example, a business policy may requires that different classes of customers enjoy different maximum resolutions as is currently the case for Netflix users, see Section 5 for more detail discussion of business policies.

4.1 The LV-based Resource Allocation

An ABR streaming session typically starts at the lowest video rate when the buffer is empty, with the ultimate aim of ramping-up the video rate to the maximum available bitrate by the time the buffer is filled [9, 20]. A series of state traversals that move the system from the initial state to a target state defines the system trajectory, and henceforth is called the *rate map*. When the $R_{max} < C(t)$ and we have no other competing clients, the rate map easily converges at the R_{max}. However, this case may not be always obtainable, as capacity can be limited and competition for it, is inevitable. To arbitrate bandwidth consumption amongst competing clients sharing a network bottleneck, we assume that the video delivery service is analogous to a biological ecosystem. We are motivated by the deep similarity that population dynamics shares with adaptive streaming system. To help us better elucidate this matter, let us begin by assuming that the video bitrate is the unit of population whose growth we are interested in. As in the case of modelling the biological system, the bitrate of the downloaded chunk represent our system state (Section 3, Assumption(1)). Furthermore, let us presume that the playback buffer is its habitat, which as in the population system, is limited and limits the maximum size of both the number of chunks we can cache and the downloaded bitrate of the chunks (Section 3, Assumption (2)).

We begin with the case where a client starts streaming at the minimum video rate, which as in the natural system represents the seed population (Section 3, Assumption (3)). After the chunk $i > 1$ is received, the client requests chunk $i+1$ with the video rate such that $x_{i+1} = rx_i \leqslant R_{max}$. The download of additional chunks is undertaken back-to-back except when the buffer is full. This ensures that the periodic (ON-OFF) chunk request policy, generally associated with adaptive streaming, is only used to prevent buffer overflow. By so doing, TCP is allowed to behave as designed. Additionally, an increase in video rate is assumed to be density dependent (Section 3, Assumption (4)). That is, the higher the video rate, the less we are inclined to increase. A number of studies [7,20] have shown that 1) users are not keen on increasing video rate when the current video rate is already sufficiently high, and 2) the higher the video rate, the greater the risk of re-buffering (stalls) events occurring. Using a different formulation model from those in the literature [22], we can now formulate a single streaming service as follows:

$$\frac{dx_i}{dt} = rx_i\left(1 - \frac{x_i}{C}\right). \tag{2}$$

In this context, clients compete for a given resource, which in this case is the available bandwidth. First, let us assume that users who sign the same SLA are to be accorded similar treatment. Therefore, it is logical to consider all flows from their streaming devices to be analogous to animals of the same species. Henceforth, we say they belong to the same class. However, if users signed different SLA they are considered to belong to different classes, because they have different demands and require different treatment. Next, we assume the data flows initiated by HAS clients compete with each other for the available bandwidth. The competitions, between flows originating from devices belonging to the same class, are governed by the intra-species coefficient β. And competitions between flows originating from devices belonging to different classes are controlled by the inter-species coefficient α. Simply put, the flow rate of each client is affected by the presence of both clients belonging to a similar and different class. Hence, Eq 2 is reformulated, to the form of Eq 1, and presented as follows:

$$\frac{dx_i}{dt} = x_i\left(r_i - \frac{\beta_i r_i}{C} - \frac{r_i}{C}\left(\sum_{j=1}^{n}\alpha_{ij}x_j\right)\right). \tag{3}$$

For convenience, henceforth, Eq 3 is rewritten in matrix form [2,29,30], as follows

$$\frac{dx}{dt} = X \circ (R - Ax). \tag{4}$$

Where $X = diag(x_1, x_2, x_3, ..., x_j)$ is an $j \times j$ dimensional matrix, $\circ$ is element-wise multiplication, $R = (r_1, r_2, r_3, ..., r_i)^T$ is the i-dimensional vector of all intrinsic growth rate constants, $x = (x_1, x_2, x_3, ..., x_i)^T$ is an i-dimensional vector of all the clients' flow rates and A is a metric shown in Eq. 5:

$$A = \frac{1}{C}\begin{bmatrix} \beta_1 r_1 & \alpha_{12}r_2 & \alpha_{13}r_3 & ... & \alpha_{1j}r_i \\ \alpha_{21}r_1 & \beta_2 r_2 & \alpha_{23}r_3 & ... & \alpha_{2j}r_i \\ \alpha_{31}r_1 & \alpha_{32}r_3 & \beta_3 r_3 & ... & \alpha_{3j}r_i \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_{i1}r_1 & \alpha_{i2}r_2 & \alpha_{i3}r_3 & ... & \beta_i r_i \end{bmatrix}. \tag{5}$$

The solution of Eq. 3, as obtained in [30], assuming a client starts streaming with $x_i = x_i(0) \geq R_{min}$, is:

$$x_i(t) = \frac{w_i(t)x_i(0)}{\beta_i x_i(0) + [w_i(t) - \beta_i x_i(0)]\mathrm{e}^{\frac{w_i(t)r_i}{C}\Delta t}}. \tag{6}$$

$$w_i(t) = C - \sum_{j\neq 1}^{n}\alpha_{ij}x_j. \tag{7}$$

From Eq. 6, the following three important facts can be deduced: First, that r_i dictates the convergence time, which is the time it takes a player to reach the steady state; Second, β_i determines the maximum converged video rate, that is the highest video bitrate that a player converges at and finally, α_i determines the maximum instantaneous target video rate.

5 System Behaviour

In this section, we investigate the dynamic behaviour of the derived model and the conditions under which a fair and stable equilibrium can be achieved.

5.1 Equilibrium and Stability

We begin by investigating the equilibrium points of the system. An equilibrium point of the proposed model represents a stationary condition. In other words, it defines the converged state that corresponds to a constant streaming condition, for a given streaming client. We find the equilibrium points of Eq. 4, by setting the equation to zero, thus the equation becomes

$$X^* \circ (R - Ax^*) = 0. \tag{8}$$

The solution will result in 2^i combinations of zero and non-zero equilibrium points [13]. This represents the states at which some or all clients either fail or survive the competition for the network resource. Let H be the set of all clients that are out-competed. That is H is a subset of streaming client $I = \{1, ..., i\}$, such that $x_h^* = 0 \; \forall h \in H$. Therefore, $I = J + H$, with $x_j^* > 0 \; \forall j \in J$ being an element of the subset of the flows that survive the competition. In this paper, we are interested in finding the condition under which all clients will survive the competition. That is, when $H = \emptyset$, so such that $I = J$ and all elements of J have a positive non-zero equilibrium point.

According to [13,25], we can find the stable non-negative equilibria x^* when the matrix in Eq. 5 is positive definite. However, in general, this is difficult to determine analytically [2,13], and when I is very large it becomes computationally intractable to do an exhaustive search for the feasible points [13]. However, an approximate solution can be found in a reasonable time, if we restrict the search space. For this, we use three representative business policies [1].

5.1.1 Policy I: All Flows are Treated Equally

For this policy, all flows are accorded the same treatment, this scenario corresponds to a situation whereby a content provider is interested in treating all customers equally. In this case, we can group all clients into a single class, making $n = 1$ (n represents the number of classes). In doing so, both the intra-species and inter-species coefficients (β, α), as well as r will be equal for all flows. Therefore, $\alpha_{ij} = \alpha$, $\beta_i = \beta$ and $r_i = r$. We can now reformulate Eq. 5 as the following:

$$A = \frac{r}{C}\left[\beta \; \alpha\right]. \tag{9}$$

Eq. 8 has two sets for equilibrium points, the first is when $X^* = 0$, and the second when x^* is a set of positive non-zero stable equilibrium presented in the following equation:

$$r - A \cdot x^* = 0. \tag{10}$$

The solution of the above expression is now presented as

$$x^* = \frac{C}{\alpha(n-1) + \beta}. \tag{11}$$

5.1.2 Policy II: Device-based Differentiation

When heterogeneous devices are used for streaming, it is desirable to allocate resources in a manner that takes the specificity of the devices into consideration. Recall from Section 4, streaming devices can be differentiated using the range of bitrates they support. If we assume the range of the supported bitrates of all the streaming clients can be grouped into n classes. For the sake of demonstration, in this paper we use $n = 3$. Now, our task is to find the vector of the three

[1] It is worth noting that these policies are by no means the only policies that can be implemented, we use them only to demonstrate how the scheme can be used in practice.

non-negative stable equilibria such that $\dot{x^*}_i \leq R_{max}$ where $i \in \{1,2,3\}$. Recall from Section 4, the value β_i determines the global maximum rate of a flow, hence, the intra-species coefficient (β_i) can be used to differentiate devices belonging to different groups, while the inter-species coefficient (α) can be left the same for all devices, hence, $\alpha_{ij} = \alpha$. Furthermore, for simplicity we assume $r_i = r$ [2], and with this information we can rewrite Eq. 5 as:

$$A = \frac{r}{C} \begin{bmatrix} \beta_1 & \alpha & \alpha \\ \alpha & \beta_2 & \alpha \\ \alpha & \alpha & \beta_3 \end{bmatrix}. \tag{12}$$

The solution of Eq.10 using the the expression of A in Eq. 12 shown below, results in three non-negative stable equilibrium points (x_1^*, x_2^*, x_3^*).

$$x_i^* = \frac{C}{\alpha(n-1) + \beta_i}. \tag{13}$$

5.1.3 Policy III: Subscription-based Differentiation

For this policy, let us assume a service provider has three subscription packages: $\{1,2,3\}$ in decreasing order of priority. All the three classes of flows share the available bandwidth at the ratio $e : f : g$. To enforce this policy, we assume that the impact of inter-specific competition from higher priority class is felt by all classes lower than it, however, a higher priority class is impacted only by the class next to it in the priority list. This corresponds to a scenario, in nature, where the impact of a powerful species is felt by all other less powerful species, but in turn the powerful species is only impacted by a competitor of comparable strength. This prevents any class from dominating others. Since the allocation is independent of the nature of the streaming devices $\beta_i = \beta$ and $r_i = r$. Therefore:

$$A = \frac{r}{C} \begin{bmatrix} \beta & \alpha_{12} & \alpha_{13} \\ \alpha_{21} & \beta & \alpha_{23} \\ 0 & \alpha_{32} & \beta \end{bmatrix}. \tag{14}$$

The solution of Eq. 10, considering A from Eq. 14, yields:

$$\begin{aligned} x_1^* &= \frac{C(\alpha_{12} - \beta)}{(\alpha_{21}\alpha_{12}) - \beta^2}, \\ x_2^* &= \frac{C(\alpha_{22} - \beta)}{(\alpha_{12}\alpha_{21}) - \beta^2}, \\ x_3^* &= \frac{C}{\beta}\left[\frac{1 - (\alpha_{32}(\alpha_{21} - \beta) - \alpha_{31}(\alpha_{12} - \beta))}{\alpha_{21}\alpha_{12} - \beta^2}\right]. \end{aligned} \tag{15}$$

[2] For this policy, there are several ways one can formulate the expression of $\dot{x^*}_i$, here we only present the simplest formulation we can derive.

5.2 Convergence and Parameter Selection

Our desire is to have each client converge at an SLA determined video rate. For this to happen, we have to ensure that the stable equilibrium points obtained in Eq. 11, 13 and 15 correspond to the desirable solution. However, as can be seen from these equations, the result of our solutions depend on the values of three constants: r, α, β, and the available bandwidth.

Like in any adaptive streaming service, any factor that influences the ability to accurately estimate the available capacity C, such as TCP congestion control, chunk size and throughput estimation methods, will certainly affect the the value of x^*. While the criterion for choosing a specific TCP protocol is outside the scope of this paper, our earlier policy to request video chunk back-to-back decouples the TCP control loop from the ABR dynamics [9,21], and enhances the ability of TCP to more easily reach steady state. When estimating the available throughput we suggest that forecasting is undertaken on aggregate flows, as we found this method achieves a better estimate. Something worth highlighting at this point is that the converged video rate is independent of streaming and content characteristics, provided they have no influence on our ability to estimate the available network capacity. Factors such as finite video lengths and different streaming lengths have no influence on x^* as can be seen its previous formulations .

5.2.1 Growth constant (r)

The value of r determines the rate at which the video rate changes with respect to time, independent of any competition. If we simplify Eq. 2 to $\frac{dx_i}{dt} = ru$, where $u = \left(1 - \frac{x_i}{C}\right)$, it is easy to see that r dictates the convergence time. Too high a value of r makes the system overly aggressive and too a low value makes the system less responsive. To ensure that the system converges in a reasonable time, we require r to be a function of buffer size. This helps ensure that the selected value of r is buffer size dependant. First, it should be noted that, generally in HAS services, buffer occupancy is measured in units of time. Therefore, we can conveniently replace t with B_t. We assume streaming begins, for all devices irrespective of policy or class flow, with a flow rate of $x(0)$ and reaches $x(B_{max}) \geq R_{max}$ when the buffer is full. Using this information we can derive the expression of r from Eq. 6 and visualise it in Figure 1 as:

$$r_i = \frac{R_{max}}{B_{max}C} \ln \left[\frac{R_{max}x(B_{max}) - x(B_{max})x(0)}{R_{max}x(0) - x(B_{max})x(0)} \right]. \tag{16}$$

5.2.2 Intra-species (α) and Inter-species(β)

For the remaining two parameters (α and β), we cast the solution into a single-objective multi-constraint optimisation problem that maximises the sum of the converged video rates:

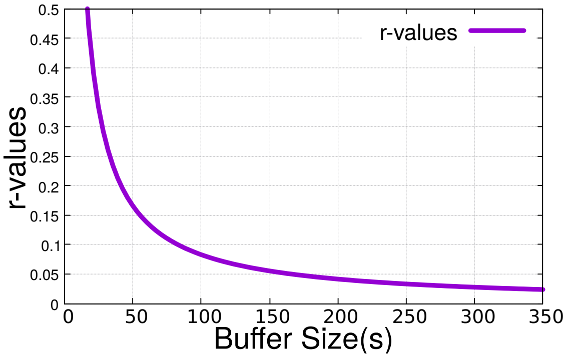


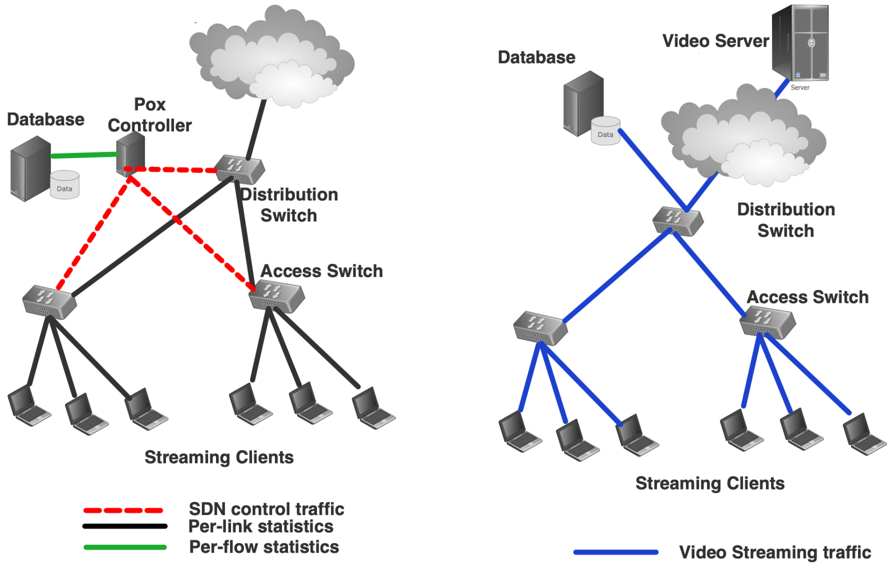
Fig. 1: The mappings between values of r to buffer levels

$$\begin{aligned} \text{Maximise:} \quad & F = \sum_{i=1}^{i} x_i^*, \\ \text{Subject to:} \quad & F \leq C, \\ & \beta_i > \alpha_{ij}, \\ & \beta_i \geq 1, \\ & x_i^* \leq R_{max}. \end{aligned} \tag{17}$$

Where $x_i^* > 0$ is the converged video rate of flows from class i at the stable equilibrium, as derived in the previous section. To further restrict the solution, the objective function, in Eq. 17, is restrained by three constraints. The first restricts the sum of all flow rates to, at most, the maximum available capacity of the bottleneck link; the second ensures that flows belonging to a class that can survive in the presence of flows from other classes (for more details see [2]); and the third ensures that the video rate selection function does not converge at a video rate greater than the allocated share. It should be noted that our formulation is network architecture independent. The only topology depends variable is the capacity of the bottleneck link, which can be located anyway within a network.

6 System Implementation and Experimental Setup

To evaluate the proposed scheme, which we call *Bio* henceforth, we envisioned a scenario where a number of geographically distributed customers are connected to a back-end server via a single network slice. Within each location, a number of streaming devices, subscribing to a managed video service, are connected via a single access switch, which is in turn connected to the Internet via a distribution switch. To facilitate better understanding, Fig. 2 presents the schematic diagram of our experimental set-up decoupled into the control and monitoring traffic plane (Fig. 2a) and the video streaming traffic flow plane (Fig. 2b). Our test-bed is inspired by the work of [5].



(a) Control and monitoring traffic (b) Video traffic

Fig. 2: Experimental Set-up

Table 2: Video resolutions and the supported Bitrate

Screen Resolution	Video Bitrate (kbps)
1080p	100-4500
720p	100-2300
360p	100-1000

As can be seen from fig. 2a, the test-bed is an SDN-enabled infrastructure. All streaming clients used are Mininet [3] instances connected via software-based OpenFlow-enabled switches. All switches are connected to a Pox controller [4]. In the test-bed, the link between the distribution switch and the video server is set as the bottleneck with a total bandwidth allocation of C. The value of C is scenario dependent, while all other links are set to 100 Mbps. This ensures that contention happens only at the bottleneck link.

MongoDB, [5], a highly scalable NoSQL database management system, is used for all data archival. To allow for maximum portability, the Pox controller and the MongoDB are hosted on the native machine. All communications with the MongoDB are done via REST calls. Throughout our implementation, the PyMongo [6] is used to handle REST calls. As can be seen in Fig 3 Host Tracker, Measurement

[3] http://mininet.org/
[4] https://github.com/noxrepo/pox
[5] https://www.mongodb.com/
[6] https://api.mongodb.com/python/current/

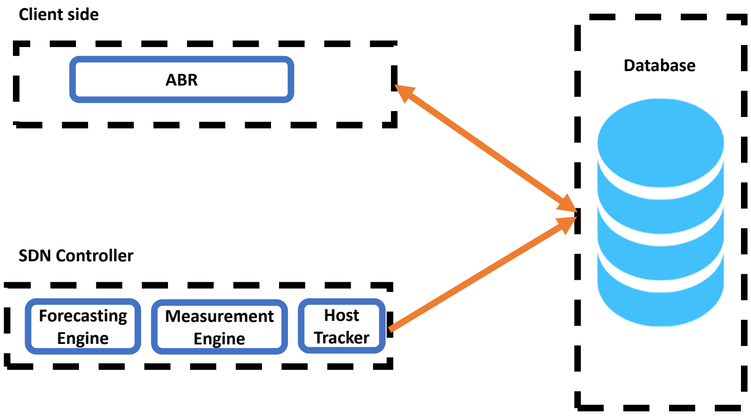


Fig. 3: Distribution of all the implemented modules.

Engine and Forecasting Engine were all implemented on top of the SDN controller. The *Host Tracker* monitors and updates the database with the current activity status of all streaming clients (i.e. whether a client is active or not). It also registers new clients to the service and then raises an event that informs the *Measurement Engine* to begin measuring the throughput of the flows of the new client, thus offering a highly dynamic and scalable framework that accommodates new users as well as optimising delivery for existing users. The measurement engine relies on a modified version of OpenNetMon [27], a network monitoring module used to measure the per-flow throughput of the video streams. The output of the measurement engine is stored in the database. Since the measurement is historical in nature, we include a *Forecasting Engine*. This module takes both per-flow and per-link measurements produced by the measurement engine, and predict the future throughput at flow and link level. For the forecast, we compute the exponentially weighted moving average (EWMA) for both per-flow and per-link throughput. A weighting factor of 0.9 is used in both instances.

Fig. 2b demonstrates how the video streaming traffic flows through the test-bed. Within each Mininet instance, we implemented a Python-based DASH compliant headless player. By disabling the decoding and rendering capacities, we are able to significantly reduce our CPU footprint without affecting the player's network behaviour. Algorithm 1 presents the detail of the ABR algorithm implemented in the headless player. A Lighttpd [7], hosted on a separate machine, is used as the video server and hosts the DASH video content. To create a path from the streaming clients to the server, the machine hosting the SDN infrastructure is equipped with two network interfaces, with one of the interfaces is connected to the distribution switch and the other interface is connected a physical switch that has been connected to the video server. For the video content, we use the Big Buck Bunny video data-set with a 2-second chunk size [11]. Detail of the three

[7] https://www.lighttpd.net/

resolutions and their respective range of bit-rates is shown in Table 2. The player is composed of two threads, the first thread, which is fired after a successful chunk download, requests from the database the latest capacity of the shared link, the client's share and updates it with a new value of the client's current download rate. The second thread uses the data from the first thread to compute the video rate of the next chunk using Eq. 6. The player uses the Python *Request* package for HTTP request-response transactions.

We compare our proposed service against using MPD modifications to restrict the maximum video representation a client is allowed to download. In line with this, two other HAS algorithms are implemented, henceforth called the *baseline players*. These are a Buffer-based player [9] called *BBA* and a hybrid player [16] called *Hybrid*. It should be noted that our desire, in this evaluation, is to find the extent to which our proposed scheme is able enforce convergence at a business policy determined rate in a stable manner. Hence, for the evaluation, we first assess how close the maximum and average video rates of both clients are to the policy defined maximum video rate. Then we compare the performance of the baseline players against our proposed scheme using the following metrics, which are widely recognised as the key QoE indicators: video bitrate instability, calculated as the percentage of video rate changes; network utilisation, calculated by dividing the total average video rate achieved by all clients by the average network capacity; and the perceptual video quality computed using the function developed in [6] for mapping video rate to Structural Similarity Index (SSMI). Each experiment is conducted five times and the average is presented. For finding the appropriate parameters of the policies outlined in Section 5.2. We use the APmonitor Optimization Suite [8]. APMonitor is a language for mixed-integer and differential algebraic equation modelling.

7 Experimental Evaluation

This section presents the results of the evaluation of policies modelled in Section 4. Before each experiment, we solve the optimisation in Eq. 17 for the values of r, α_i, β_i. On a MacBook Pro (memory: 16GB, Processor: 3 GHz) each round of the optimisation took on average 30sec to finish. We feel this is acceptable for real-time usage, especially since we do not expect business policies to change very often.

7.1 Policy I

This policy is to guarantee a 2Mbps video rate for all streaming devices. For this test, we vary the number of streaming clients (4, 9, 16, 25) and the respective bandwidth of the bottleneck link (8Mbps, 18Mbps, 32Mbps, 50Mbps). A new player is introduced every 5 seconds. Each player streams using the 1080p video set (see Table 2). The following parameters are obtained by running the optimisation: $r = 1, \beta = 1.77, \alpha = 0.76$. Furthermore, the initial rate ($x_{s_i}(t = 0)$) is set to the minimum video rate. To properly evaluate the performance of the proposed scheme, we compare it against two scenarios: Fig. 4: when the baseline players are

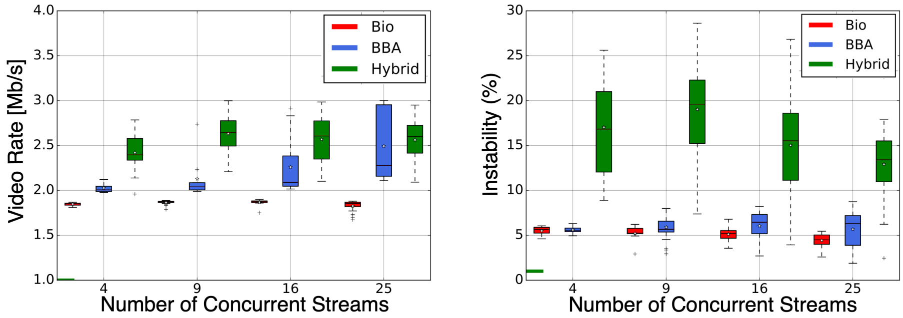
[8] www.apmonitor.com

Algorithm 1: Client-side Algorithm

```
input  : q_n
         B_t
output: q_next
         B_delay
         x_s * α
if B_t := 0 then
    B_delay = 0
    q_next = q_0
    if get_link_capacity() ⩽ 0 then
        C = rand
        w(t) = C
    else
        C = get_link_capacity()
        w(t) = rand
    end
else
    if U(t+1) ⩽ q_{n-1} then
        q_next = q_{n-1}
    else if U(t+1) ⩾ q_{n+1} then
        q_next = q_{n+1}
    else
        qnext = q_n
    end
    if q_n =: q_0 then
        q_next = q_0
    end
    if q_n =: R_max then
        q_n = R_max
        B_delay = V
    end
    q_n = q_next
end
```

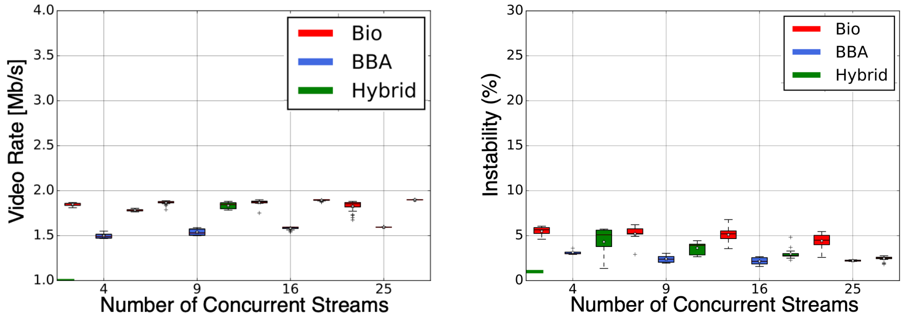
free to choose any video rate in the MPD and Fig. 5: when the baseline players are limited to the video rates in the MPD below a specific threshold. As can be seen that in Fig. 4a all the streaming clients using *Bio*, achieve an average video rate of 1.85 Mbps and converge at exactly 1.9 Mbps, which is the closest available video rate less than or equal to the requirement set by the policy. Furthermore, it can be seen the average video rate of the *Bio* algorithm is consistent regardless of the number of competing clients. However, for the baseline players, the greater the number of streaming clients the more the variation in the achieved average video rate. This occurs because as we increase the number of streaming clients per run, the overall capacity of the bottleneck link also increases. Since the clients join the network at different times, we find the baseline players that join early utilise as much capacity as possible, however, this first-come-first-served allocation disadvantages the players joining later, which ultimately leads to increased variation in the achieved average video rate.

From Fig. 4b, we note a marked increase in instability of the baseline players as the number of streaming clients per run is increased. Hybrid players suffer video rate instability of between 5% to 28%. The clients streaming using the *Bio* scheme experience the least instability, 5%, with a consistent rate regardless of the number of streaming clients.



(a) Average video rate (b) Instability Test

Fig. 4: Policy I: when HAS clients, BBA and Hybrid, can choose any available bitrate in the MPD file



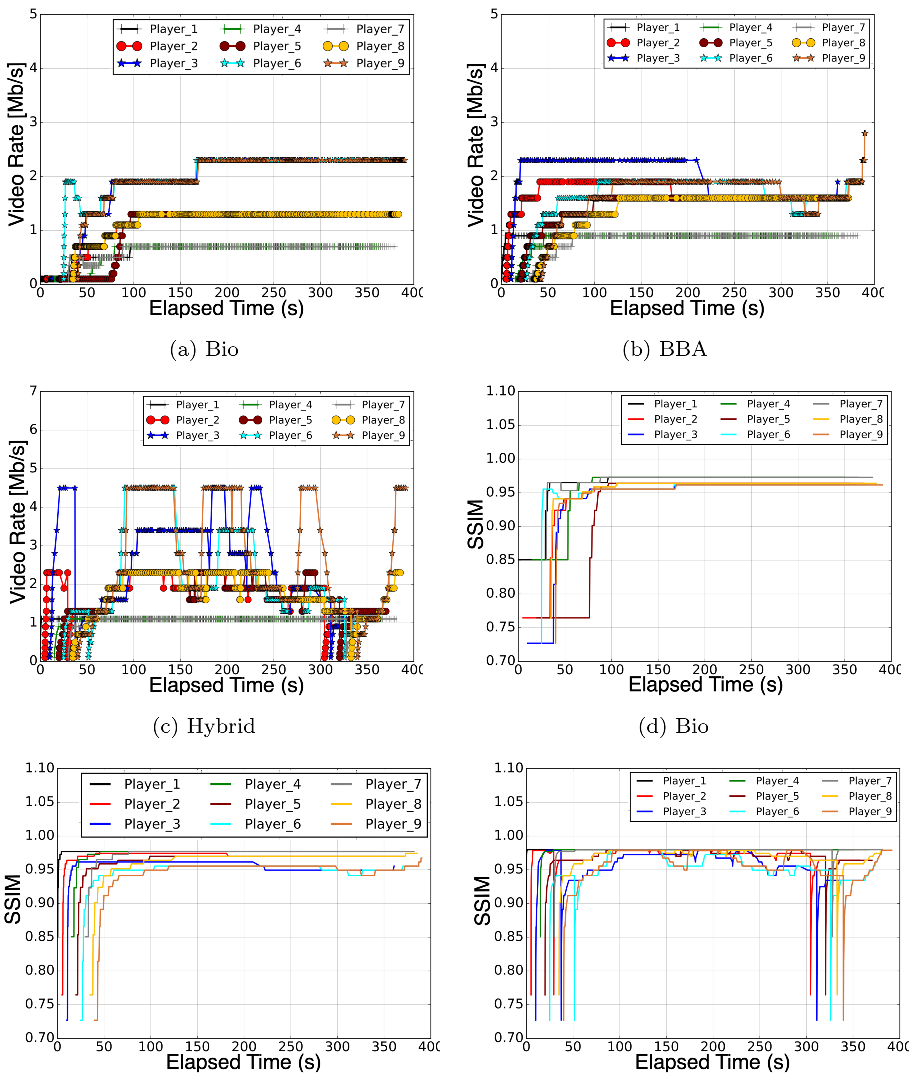
(a) Average video rate (b) Instability Test

Fig. 5: Policy I: when HAS clients, BBA and Hybrid, are limited to the same available bitrate as our bio-clients

Fig. 5 illustrates results when the baseline players are restricted to a maximum video rate of 2Mbps. In our evaluation, this is achieved by removing the higher bitrate representations in the MPD text file at the HTTP server. As can be seen in Fig. 5a, *BBA* achieves the lowest average video rate, in other words, it is the furthest from the requirement of the policy. While both *Bio* and *Hybrid* achieve an average video rate of circa 1.9Mbps regardless of the number of concurrent player per session. This is achieved with a high degree of stability in video rate, with all the player suffering a video rate instability of between $3 - 5\%$ (see Fig 5b).

7.2 Policy II

Policy II requires streaming devices to be differentiated based on the package they subscribed. The packages are Basic, Standard, and Premium, For this test, we run nine players and create three classes: Players 1, 4, 7 subscribe to the Basic package, which only support a resolution of 360p. Players 2,5,8 paid for the Standard package, which allows for the maximum resolution of 720p, while players 3,6,9 registered for the Premium, a package that allows for the maximum resolution of 1080p. For the baseline players, modified MPD files were used to force them to remain within the desired bitrate ranges. The bandwidth of the bottleneck is set



(a) Bio (b) BBA

(c) Hybrid (d) Bio

(e) BBA (f) Hybrid

Fig. 6: Policy II

to 13 Mbps, which is equivalent to 56% of the total rate needed to guarantee all players stream at the highest supported video rate (respective of class).

The following are the derived parameters $r = 1.0$, $\beta_1 = 2.5$, $\beta_2 = 1.7, \beta_3 = 1.3$, $\alpha = 0.76$. As in the previous policy, all the players are configured to start at the minimum video bitrate. Fig. 6 plots the video rates of the players' downloaded chunks (top row) and the corresponding perceptual video quality (bottom row).

As can be seen in Fig 6, regardless of the type of the ABR used by a player, the players subscribing to the Basic package (1,4,7) remained within the allocated range of bitrates ($\leqslant$ 1.1 Mbps). However, when the baseline players are used, the devices subscribing to either Standard or Premium package violated their contract. As can be seen in Fig. 6e, for both classes of the subscribers, when using BBA they eventually converged at the same video rate. This results in players subscribing to the Premium plan converging at lower than expected video rate and the opposite for the standard players that converged at higher rate. Consequently, the consumers that pays lower enjoyed higher perceptible video quality. In Fig. 6f, the Hybrid players made some attempt to discriminate the two classes of the players at the cost of an increase in instability and significant fluctuation in perceptible video quality.

However, when the *Bio* scheme is utilised, the three classes converged at 2.3 Mbps, 1.3 Mbps and 0.7 Mbps which is equivalent of 52%, 56% and 63% of the requirement (see Fig. 6a), this illustrates that our scheme is able to appropriately discriminate the flows, as can be seen in Fig. 6d, and results in the players enjoying the differentiated level of perceptible video quality, with a marked reduction in instability.

7.3 Policy III

Policy III requires that the available capacity is shared amongst competing flows in a proportion set by a service provider. Here in addition to being able to proportionately share the available capacity, we expect high utilisation from the competing clients. It worth noting that we deliberately do not include the SSIM plots, for it is our view that such metric is not the most suitable for evaluating bandwidth utilisation. As laid out in Section 5.1.3, category 1, 2 and 3 are assigned a decreasing order of priority. For evaluation, we set a client allocation ratio of 5:3:2. The calculated parameters are $r = 1$, $\alpha_{3,1} = 0.1$, $\alpha_{2,1} = 1.3$, $\alpha_{1,2} = 0.14$ $\alpha_{3,2} = 1.6$ and $\beta = 2.0$. As in the previous policy, the bottleneck link is set to 13 Mbps and the players start streaming from the lowest video rate. As can be seen in Fig. 7 video rates converge at a stable equilibria of 1.9Mbps, 1.5Mbps and 0.9Mbps, resulting in the three classes of flows achieving average bitrates of 1.88, 1.34 0.78 , and achieving approximately 4.4:3:1.8 bandwidth distribution ratio corresponding to 94% bandwidth utilisation. For the baseline players, we find that they cannot differentiate between the assigned allocation ratios, and the result for Policy III is identical to the one shown in Fig 4 for Policy I.

7.4 System Responsiveness

To test the responsiveness of the *Bio* model to a sudden change in the available resources, we investigated the following cases: 1) when system capacity suddenly increases either because the service provider has decided to purchase more capacity, or 2) as a result of some players leaving and 3) how the scheme responds to a sudden drop in link capacity. The plots shown in Fig. 8a to Fig. 8c present the achievable video rate (y-axis left hand side), elapsed time (x-axis) and available bandwidth (y-axis right hand side).

From Fig. 8a and 8b it can be seen both sessions started with a shared bandwidth capacity of 12 Mbps and the players converge at a stable equilibrium points of 1.9Mbps, 1.3Mbps and 0.9Mbps, corresponding approximately to a ratio of

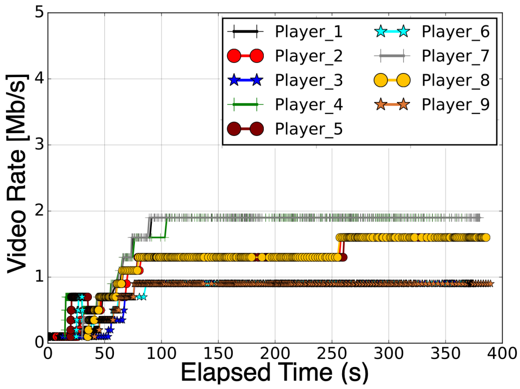
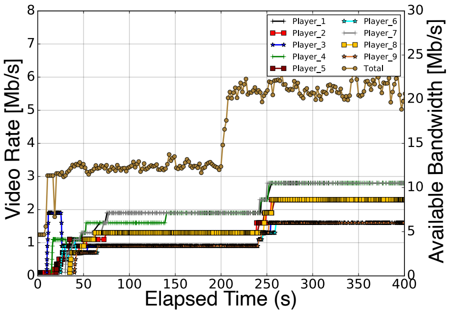
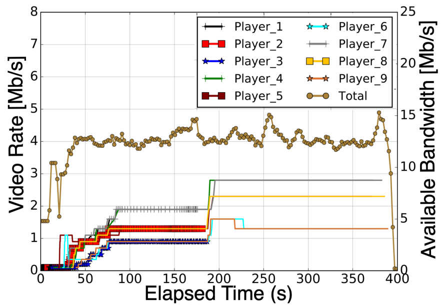


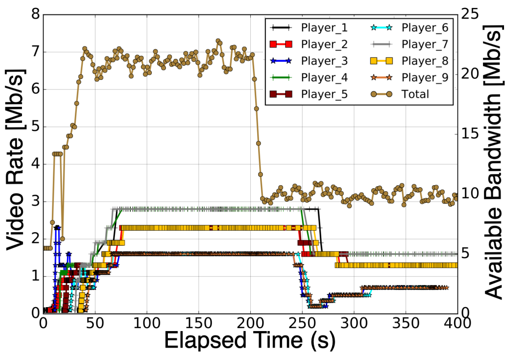
Fig. 7: Policy III



(a) Increase in Capacity



(b) Players Leaving



(c) Drop in capacity

Fig. 8: System Responsiveness

5:3:2, which is the set target of policy III. Half-way through the streaming session: for Fig. 8a we increase the shared capacity to 22 Mbps (see the brown plot) and the players switched to a new converged rate of 2.7Mbps, 2.3Mbps and 1.5Mbps (4.2:3.5:2.3). While in Fig. 8b three players (1,2,3 - one player from each class) left the session, thus increasing the available capacity by about one third for the remaining streaming clients. As can be seen, our model is able to redistribute the new capacity at a proportion required by the underlying policy. Finally in Fig. 8c we reversed the available bandwidth rates allocated in Fig. 8a, such that 22Mbps is available bandwidth at the beginning of the session, with a reduction of 12Mbps occurring half-way through the session. Similar to the earlier tests, the Bio clients smoothly changed representation rate selection with very little overall instability.

8 Conclusion

Guaranteeing an agreed SLA to a managed HTTP-based adaptive streaming service users is a challenging task. In this paper, we have introduced a distributed framework that allows a service provider to enforce business policies without the need for an active network agent. The proposed framework utilises the ecological models of species competition for a limited resource. Using SDN, we implement the framework and through careful parameter selection of inter-species and diff-species coefficients, we developed and evaluated a number of policies. Our evaluation shows that policy enforcement results in a fair, stable and high-quality video streaming service for all players. For future work, we will focus on implementing additional SLA policies and further scaling the implementation to encompass additional players, while also investigating the relocation of the bottleneck link(s) to air-interface last-hop models such as WIFI, LTE and 5G.

Acknowledgements This publication emanates from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant 13/IA/1892, and also acknowledges the support of CONNECT the SFI research centre for future networks and communications.

References

1. Guidelines for implementation: Dash-if interoperability points. Standard, DASH Industry Forum (2018)
2. Antoniou, P., Pitsillides, A.: A Bio-inspired Approach for Streaming Applications in Wireless Sensor Networks Based on the Lotka–Volterra Competition Model. Computer Communications **33**(17), 2039–2047 (2010)
3. Ashby, W.R.: Principles of the self-organizing system. In: Facets of systems science, pp. 521–536. Springer (1991)
4. Bhat, D., Rizk, A., Zink, M., Steinmetz, R.: Network assisted content distribution for adaptive bitrate video streaming. In: Proceedings of the 8th ACM on Multimedia Systems Conference, MMSys'17, pp. 62–75. ACM, New York, NY, USA (2017). DOI 10.1145/3083187.3083196. URL http://doi.acm.org/10.1145/3083187.3083196
5. Bhat, D., Rizk, A., Zink, M., Steinmetz, R.: Sabr: Network-assisted content distribution for qoe-driven abr video streaming. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM) **14**(2s), 1–25 (2018)
6. Georgopoulos, P., Elkhatib, Y., Broadbent, M., Mu, M., Race, N.: Towards network-wide qoe fairness using openflow-assisted adaptive video streaming. In: Proceedings of the 2013 ACM SIGCOMM Workshop on Future Human-centric Multimedia Networking, FhMN '13, pp. 15–20. ACM, New York, NY, USA (2013). DOI 10.1145/2491172.2491181. URL http://doi.acm.org/10.1145/2491172.2491181

7. Georgopoulos, P., Elkhatib, Y., Broadbent, M., Mu, M., Race, N.: Towards network-wide qoe fairness using openflow-assisted adaptive video streaming. In: FhMN '13, pp. 15–20 (2013)
8. Huang, T.Y., Handigol, N., Heller, B., McKeown, N., Johari, R.: Confused, timid, and unstable: Picking a video streaming rate is hard. In: Proceedings of the 2012 Internet Measurement Conference, IMC '12, pp. 225–238. ACM, New York, NY, USA (2012). DOI 10.1145/2398776.2398800. URL http://doi.acm.org/10.1145/2398776.2398800
9. Huang, T.Y., Johari, R., McKeown, N.: Downton Abbey Without the Hiccups: Buffer-based Rate Adaptation for HTTP Video Streaming. In: Proceedings of the 2013 ACM SIGCOMM workshop on Future human-centric multimedia networking, pp. 9–14. ACM (2013)
10. Kleinrouweler, J.W., Meixner, B., Cesar, P.: Improving video quality in crowded networks using a dane. In: Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video, pp. 73–78 (2017)
11. Lederer, S., Müller, C., Timmerer, C.: Dynamic adaptive streaming over http dataset. In: Proceedings of the 3rd Multimedia Systems Conference, MMSys '12, pp. 89–94. ACM, New York, NY, USA (2012). DOI 10.1145/2155555.2155570. URL http://doi.acm.org/10.1145/2155555.2155570
12. Li, Z., Zhu, X., Gahm, J., Pan, R., Hu, H., Begen, A.C., Oran, D.: Probe and adapt: Rate adaptation for http video streaming at scale. IEEE Journal on Selected Areas in Communications **32**(4), 719–733 (2014)
13. Lischke, H., Löffler, T.J.: Finding all Multiple Stable Fixpoints of n-Species Lotkaa-Volterra Competition Models. Theoretical Population Biology **115**, 24 – 34 (2017)
14. Liu, C., Bouazizi, I., Gabbouj, M.: Rate adaptation for adaptive http streaming. In: Proceedings of the Second Multimedia Systems Conference, MMSys '11, pp. 169–174. ACM, New York, NY, USA (2011). DOI 10.1145/1943552.1943575. URL http://doi.acm.org/10.1145/1943552.1943575
15. Mao, H., Netravali, R., Alizadeh, M.: Neural adaptive video streaming with pensieve. In: Proceedings of the Conference of the ACM Special Interest Group on Data Communication, pp. 197–210 (2017)
16. Miller, K., Quacchio, E., Gennari, G., Wolisz, A.: Adaptation Algorithm for Adaptive Streaming over HTTP. In: 2012 19th International Packet Video Workshop (PV), pp. 173–178 (2012)
17. MPEG: Information technology – dynamic adaptive streaming over http (dash) – part 5: Server and network assisted dash (sand) (2017). URL https://www.iso.org/standard/69079.html
18. Pham, S., Heeren, P., Silhavy, D., Arbanowski, S.: Evaluation of shared resource allocation using sand for abr streaming. In: Proceedings of the 10th ACM Multimedia Systems Conference, pp. 165–174 (2019)
19. Quinlan, J.J., Zahran, A.H., Ramakrishnan, K., Sreenan, C.J.: Delivery of Adaptive Bitrate Video: Balancing Fairness, Efficiency and Quality. In: 2015 IEEE International Workshop on Local and Metropolitan Area Networks (LANMAN), pp. 1–6. IEEE (2015)
20. Sani, Y., Mauthe, A., Edwards, C.: Modelling video rate evolution in adaptive bitrate selection. In: 2015 IEEE International Symposium on Multimedia (ISM), pp. 89–94. IEEE (2015)
21. Sani, Y., Mauthe, A., Edwards, C.: Adaptive Bitrate Selection: A Survey. IEEE Communications Surveys Tutorials **19**(4), 2985–3014 (2017)
22. Sani, Y., Mauthe, A., Edwards, C., Mu, M.: A Bio-inspired HTTP-based Adaptive Streaming Player. In: 2016 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 1–4. IEEE (2016)
23. Sodagar, I.: The MPEG-DASH Standard for Multimedia Streaming Over the Internet. MultiMedia, IEEE **18**(4), 62–67 (2011)
24. Spiteri, K., Urgaonkar, R., Sitaraman, R.K.: Bola: Near-optimal bitrate adaptation for online videos. In: IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, pp. 1–9. IEEE (2016)
25. Takeuchi, Y., Adachi, N.: Stable Equilibrium of Systems of Generalized Volterra Type. Journal of Mathematical Analysis and Applications **88**(1), 157–169 (1982)
26. Tsoularis, A., Wallace, J.: Analysis of Logistic Growth Models. Mathematical biosciences **179**(1), 21–55 (2002)
27. Van Adrichem, N.L., Doerr, C., Kuipers, F.A.: Opennetmon: Network Monitoring in Openflow Software-defined Networks. In: Network Operations and Management Symposium (NOMS), 2014 IEEE, pp. 1–8. IEEE (2014)

28. Verhulst, P.F.: Notice on the Law that the Population Follows in its Growth. Correspondence of Mathematical and Physical **10**, 113 –121 (1838)
29. Wu, K.D., Liao, W.: On service differentiation for multimedia traffic in multi-hop wireless networks. IEEE transactions on wireless communications **8**(5), 2464–2472 (2009)
30. Yao, X.W., Wang, W.L., Yang, S.H., Cen, Y.F.: Bio-inspired Self-adaptive Rate Control for Multi-Priority Data Transmission over WLANs. Computer Communications **53**, 73–83 (2014)
31. Yin, X., Jindal, A., Sekar, V., Sinopoli, B.: A control-theoretic approach for dynamic adaptive video streaming over http. In: Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, pp. 325–338 (2015)
32. Zahran, A.H., Quinlan, J.J., Ramakrishnan, K.K., Sreenan, C.J.: SAP: Stall-Aware Pacing for Improved DASH Video Experience in Cellular Networks. In: Proceedings of the 8th Multimedia Systems Conference, MMSys'17, pp. 13–26. ACM, New York, NY, USA (2017). DOI 10.1145/3083187.3083199. URL http://doi.acm.org/10.1145/3083187.3083199
33. Zhu, C., Yin, G.: On competitive Lotka–Volterra Model in Random Environments. Journal of Mathematical Analysis and Applications **357**(1), 154–170 (2009)