Incentives Against Hidden Action in QoS Overlays

Raúl Landa, Miguel Rio, David Griffin, Richard Clegg, Eleni Mykoniati
Department of Electronic and Electrical Engineering
Networks and Services Laboratory
{rlanda, mrio, dgriffin, rclegg, emykoniati}@ee.ucl.ac.uk

Abstract

Peer-to-peer networks providing QoS-enabled services are sensitive to hidden action *situations, where the actions of a server peer are hidden from the peers who receive services from it. This is because server peers can choose to strategically minimize their effort, and client peers may be unable to distinguish between cases where the server exerted insufficient effort and cases where the server kept its advertised effort levels but the end-to-end conditions in the network were sufficiently adverse.*

We propose a principal-agent *model for hidden action that gives server peers sufficient incentives to meet their advertised effort levels, without client peers having to decide for each transaction whether the outcome was due to server behavior or network conditions. This allows peers to draft contracts that provide incentives for truthful revelation of QoS capabilities, and to have predictable transaction quality. We then exemplify the model for the case of a mesh-based, pull-oriented streaming system with low delay requirements. For this case, we show how to estimate the model parameters a function of the prevailing network conditions, and how to enforce contract fulfillment through a reciprocative strategy.*

1. Introduction

The design of QoS overlays over best-effort networks is still an open research problem [29, 33, 21]. In this paper, we approach the problem of QoS as an observable indicator of peer behavior: although the peer-experienced QoS will depend on the underlay network conditions, it will be also affected by the behavior of the serving overlay nodes. We seek to propose an incentives mechanism that provides an incentive for server peers to deliver their advertised levels of effort, even if client peers are unable to unequivocally determine if the unsatisfactory outcome of a transaction is due to insufficient server effort or to network variability. We will focus on peer-to-peer overlays with strategic, selfish peers.

Although there is a large amount of work on the economic and game theoretical modeling of peer-to-peer systems (in particular as it relates to the provision of incentives against the *free-rider problem* [1, 13, 27, 10, 9]), it is frequently assumed that peers can determine the level of server effort by observing their own experienced QoS. Based on this, peers choose the service quality that they provide to other peers as a function of the service quality these other peers have, in turn, provided to the system. Unfortunately, clients are very often unable to observe the actual server effort, since it is obscured by shifting network and peer conditions that affect transfer operations. The design for QoS contracts between peers connected through a best-effort network has been, thus, less widely studied.

The main difficulty in the analysis of peer-to-peer QoS contract design is that the effect of the behavior of the server is *externalized* to the client. This means that if a server peer fails to deliver a high-enough contribution effort, and this leads to low QoS for the client, the server does not naturally bear the negative consequences of this action. Instead, these consequences are externalized to the peer who requested the service.

Since the client is unable to directly measure the actions of the server, the fact that the server might have not delivered its advertised service quality can be only indirectly inferred. In the present paper we propose a *principal-agent* model for hidden action that gives server peers sufficient incentives to meet their advertised effort levels, without the peers having to decide for each transaction whether the observable outcome (the actual QoS linked to the transaction) was due to server behavior or network conditions. This is accomplished by allowing peers to draft contracts that provide incentives for truthful revelation of system contribution capabilities. This, in turn, helps strategic peers to obtain more predictable service levels. We propose an algorithm for the calculation of the probabilistically optimal payments on a QoS-dependent contract offering greater payment if the transaction is concluded with high QoS, and a lower payment otherwise. Although we propose a general expres-

sion for the calculation of these payments, every particular application will require its own probabilistic model of the relationship between the effort that the server puts into a transaction, the prevailing network conditions and the QoS experienced by the client. We present such a model for an specific example: a mesh-based, pull-oriented streaming system with low delay requirements.

The structure of the paper is as follows. In Section 2 we approach the general problem of hidden action in QoS overlays, and present the optimal differentiated payments that a client must offer to provide a server with the incentive to deliver its advertised level of service. In Section 3 we apply the model to a delay-sensitive chunk distribution overlay for media streaming, and show how the model parameters can be defined in terms of observable network measurements. In Section 4 we present other attempts to develop optimal peer-to-peer contracts, or to use the principal-agent model in the context of computer networking.

2. Modeling Hidden Action

The term *hidden action* is used when, in a two party transaction, the actions of one of the parties (usually regarding contract compliance or lack thereof) are hidden from the other. One model for these cases is the *principal-agent* model, where an economic actor (the *principal*) trusts another one (the *agent*) to perform a task. However, the actions that the agent takes (or fails to take) in the context of the task are not completely verifiable by the principal. Thus, the agent has the power to impose, through its actions, an externality on the principal. To transfer some of this risk back to the agent, the principal will usually require a contract by which the value that the the agent obtains from the transaction will depend on the observable consequences of its actions. This contract is designed to transfer at least part of the externality back to the agent. Additionally, this contract creates an "audit point" where the principal is able to assess the actions of the agent, and respond accordingly.

In the case of peer-to-peer QoS overlays, the client plays the part of the principal and the server plays the part of the agent: the client trusts the server to provide its advertised effort, but the experienced QoS of the interaction will also depend on the variability of the network on which the service takes place. Thus, the server will have some degree of plausible deniability if its service is unsatisfactory, as degradation might be indicative of fluctuating network conditions and not lack of effort on its side.

In order to use the principal-agent model, we must assume the existence of a *market and currency system* $\mathcal{M}$. The role of $\mathcal{M}$ includes informing all peers of the effort guarantees advertised by other peers, communicating pricing and resource availability information, securely maintaining the amount of currency that peers possess, and executing currency transfers between peers. Thus, $\mathcal{M}$ is the open market on which peers advertise their resources and formalize transactions. We assume that services are found, purchased and paid for using currency and procedures defined in $\mathcal{M}$.

There is a vast body of research regarding the distributed implementation of $\mathcal{M}$ using either structured or unstructured overlays [32, 34, 6, 31, 15, 36, 30, 38, 2], and our technique can be easily adapted to be used on any one of these systems[1]. Instead of proposing our own implementation of $\mathcal{M}$, we propose an open model applicable to a wide range of possible $\mathcal{M}$ implementations, and leave its precise modeling and analysis outside the scope of this paper.

Through $\mathcal{M}$, clients search for prospective servers according to the QoS characteristics that the servers themselves advertise. This takes the form of either a distributed hash table (DHT) query, or directed broadcast on an unstructured overlay. Of course, servers will only truthfully reveal their true QoS capabilities if the benefit they can obtain if they do so is greater that the benefit they can obtain if they lie. We will explore this issue in Section 3.

A peer i with a server role will only provide a service to a client j if the utility that a transaction with j offers is at least as large as the average utility that i could obtain from a transaction with any other peer k. Thus, if we define U_r as the average utility that a peer can expect as a result of a transaction, we have that the minimum utility that i will accept from an interaction with j is U_r (we call this the *rationality* condition).We assume that peers can query $\mathcal{M}$ to learn U_r, and that U_r is updated by $\mathcal{M}$ with every successful transaction.

Once i has accepted a request from j, it has to determine the treatment it will give to it: i might give different priority to different requests, or it might have other local processes competing for scarce CPU or upload bandwidth resources. Thus, i can respond to the request from j with various levels of *effort* $\phi \in \mathbb{R}_{\geq 0}$. The service quality $q \in \mathbb{R}_{\geq 0}$ that j experiences as an outcome of the transaction will be contingent on ϕ: higher values of q are positively correlated with higher values of ϕ.[2]

However, we assume that the client is unable to observe ϕ - this is *private information* of the server. Additionally, we assume that there is uncertainty in QoS outcomes: due to varying delay and throughput conditions in the network, it is impossible for j to infer ϕ directly from the observed transaction QoS q. Thus, an unobservable decision by i (its choice of ϕ) has an effect on the utility experienced by j,

[1] We assume that peers have reliable identities, and thus we do not address Sybil or whitewashing attacks [8, 14].

[2] We are deliberately vague in the definition of ϕ and q, as they are application-dependent. For some overlays ϕ and q might be defined in terms of delay, while for others they might be better defined in terms of jitter and average throughput. We provide a specific mapping of ϕ and q to time-sensitive chunk transfer on Section 3.

ϕ	Level of effort by the server
U_s	Utility function for the server
U_c	Utility function for the client
q	QoS enjoyed by the client
ψ	Payment given by the client to the server
p	Probability of the client getting high quality service as a function of the effort by the server.
U_r	Reservation utility for the server

Table 1. Principal-Agent model notation

and we have a situation where the principal-agent model applies [22]. The client is unable to unambiguously infer the behavior of the server from the observation of the transaction outcome, and thus is vulnerable to exploitation by it. Thus, the server needs to be given an incentive (in the form of an outcome-dependent payment $\psi \in \mathbb{R}_{\geq 0}$ measured in the currency units of $\mathcal{M}$) to give its best effort by assimilating some of the risk associated with the network conditions that might affect the transaction outcome. These variables are summarized in Table 1.

2.1. The Principal-Agent Model

The objective of the model is to obtain the payments that the client needs to offer the server as a function of the quality of service it receives, in order to ensure that the server puts maximal effort into maintaining its advertised effort levels. The client thus calculates payments as a function of the server-advertised effort (and the estimated condition of the overlay network link between them) and creates a contract for the server. The server can either accept the contract as it stands, or reject it immediately.

If the server is able to maintain the effort commitments that it advertised in $\mathcal{M}$, it is considered to have devoted *high effort* to the transaction, and $\phi = \phi_+$. On the other hand, if the server is unable to maintain its advertised effort, it is considered to have devoted *low effort*, and $\phi = \phi_-$. Of course, low levels of effort increase the chance of the outcome having low QoS, and we seek to create an incentive mechanism against them[3].

A transaction is defined to have failed if the quality that the client experiences is lower than the minimum quality that the client is willing to tolerate, as set in the initial client-server contract. Thus, for any successful transaction, the outcome for the client can be either *high QoS* (and $q = q_+$) or *low QoS* (with $q = q_-$). We address transaction failure in Section 3.1.

The outcome of the transaction (the service quality experienced by the client) depends probabilistically upon the level of effort of the server:

$$\begin{aligned} P[q = q_+ | \phi = \phi_+] &= p_+ \\ P[q = q_- | \phi = \phi_+] &= 1 - p_+ \\ P[q = q_+ | \phi = \phi_-] &= p_- \\ P[q = q_- | \phi = \phi_-] &= 1 - p_- \end{aligned}$$

We denote with ψ the payment that the server will receive after the transaction is completed. The client will pay the server a differentiated amount, according to the QoS that it experiences. Thus, the client will pay the server $\psi = \psi_+$ if the outcome is high QoS, a smaller amount ψ_- if the outcome is low QoS, and nothing if the transaction fails. As stated, $\phi_+ > \phi_-$, $\psi_+ > \psi_-$ and $q_+ > q_-$.

We continue the derivation of the optimal contract payments by defining the following utility function for the client:

$$U_c(q, \psi) = q^\beta - \psi$$

where $\beta \in (0, 1)$. Let U_c^+ be the expected utility for the client if $\phi = \phi_+$, and U_c^- the expected utility for the client if $\phi = \phi_-$. Then, we have that:

$$\begin{aligned} U_c^+ &= p_+ U_c(q_+, \psi_+) + (1 - p_+) U_c(q_-, \psi_-) \\ U_c^- &= p_- U_c(q_+, \psi_+) + (1 - p_-) U_c(q_-, \psi_-) \end{aligned}$$

The client wishes the server to put high effort in the transaction, and thus to calculate ψ_+ and ψ_- to maximize U_c^+ (we shall not consider optimizing U_c^-). We define the following utility function for the server:

$$U_s(\psi, \phi) = \psi^\alpha - \phi$$

where $\alpha \in (0, 1)$[4]. Then, the expected utility for the server as a function of its effort and payment is:

$$\begin{aligned} U_s^+ &= p_+ U_s(\psi_+, \phi_+) + (1 - p_+) U_s(\psi_-, \phi_+) \\ U_s^- &= p_- U_s(\psi_+, \phi_-) + (1 - p_-) U_s(\psi_-, \phi_-) \end{aligned}$$

We seek to ensure that the server obtains a higher utility by exerting ϕ_+ than by exerting ϕ_- (we call this the *incentive compatibility* constraint). Thus, we must find the ψ_+ and ψ_- that solve the following optimization problem:

$$\begin{aligned} \text{Maximize:} \quad & U_c^+ \\ \text{Subject to:} \quad & U_s^+ \geq U_r \quad \text{(rationality)} \\ \text{And:} \quad & U_s^+ \geq U_s^- \quad \text{(incentive compatibility)} \end{aligned} \tag{1}$$

[3] The mapping between ϕ_+, ϕ_-, q_+ and q_- and observable parameters for an example peer-to-peer streaming application is analyzed in Section 3

[4] α and β are external parameters that can be used to fit these functions to experimental measurements. For the rest of this paper, we consider them external parameters that are given.

It can be shown (see Appendix A) that the optimal payments for the outcome-dependent QoS contract are:

$$\psi_- = \left(U_r + \frac{p_+\phi_- - p_-\phi_+}{p_+ - p_-} \right)^{\frac{1}{\alpha}} \quad (2)$$

$$\psi_+ = \left(U_r + \frac{(1-p_-)\phi_+ - (1-p_+)\phi_-}{p_+ - p_-} \right)^{\frac{1}{\alpha}} \quad (3)$$

3. Case study: Delay-sensitive chunk transfer in streaming systems

In this section, we focus on modeling hidden action in a *mesh-based*, *pull-oriented* peer-to-peer media streaming system. In mesh-based systems, the media stream is divided in constant-sized *chunks* at its source, and each one of these is transported through the overlay in a more or less independent fashion. In pull-oriented systems, every chunk is negotiated in individual interactions where a request for an specific chunk is issued. We call each of these interactions a *transaction*. A transaction consists of a peer (the *client*) issuing a request to another peer (the *server*) for a chunk, and the server responding with either the chunk in question or a message refusing the request. Of course, we require no asymmetrical restriction in roles: peers can be simultaneously clients and servers to other peers. Additionally, peers freely select which peers to upload to and download from, and each chunk is individually requested through a logically separate operation. Examples of these include DONet [37], PULSE [26] or Bullet [20].

In order to simplify the analysis, we assume that signaling and stream traffic is exchanged using TCP over already open connections[5].

We assume that the servers advertise, using $\mathcal{M}$, the effort levels that they are willing to commit to. Finally, we assume that peers need to schedule chunk downloads from other peers in a predictable manner in order to be able to maximize their QoS. Thus, each client peer is assumed to solve an optimization problem to decide which servers to ask for which chunks as a function of the effort parameters that they advertise, and its local codec play-out requirements. Our problem is then to design an incentives system that ensures that the server peers respond according to their advertised effort levels.

We model a transaction as shown in Fig. 1 (the variables therein are explained in Table 2). The total delay that a client experiences on a transaction is:

$$D = t_{RTT} + \frac{S_r}{T_c} + t_P + \frac{N_s S_c}{T_s} \quad (4)$$

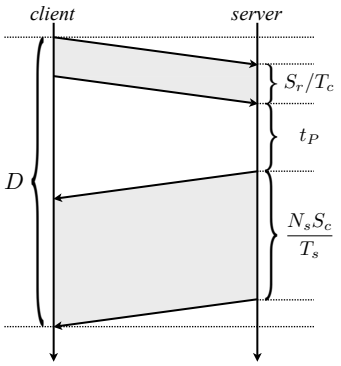

Figure 1. Transaction delay components

Clearly, the variables in Table 2 fall into two kinds: those under the control of the server (t_P and N_s) and those that are a property of the protocol specification or the TCP connection between the client and the server (t_{RTT}, S_r, T_c and T_s). Thus, it is t_P and N_s that constitute the "advertised effort levels" that are disseminated through $\mathcal{M}$.

We model the effort put by the server on a given transaction by defining two deadlines: the first one, t_+, denotes the maximum transaction resolution time if the server actually delivers its advertised effort. The second deadline, t_-, is the absolute maximum delay that the client is willing to tolerate for the transaction. If a transaction was not rejected by the server immediately after its request, the elapsing of t_- without the client having fully received the requested chunk will be interpreted by the client as deviation from a socially enforced rule, triggering penalties for the server (as detailed in Section 3.1).

In order to fully characterize the behavior of the client, we also define a time t_c during which the server has the opportunity to reject the transaction. If the client does not receive a rejection message before t_c elapses, it will silently assume that the server has accepted the request and it will deliver a chunk with the effort level that it advertised through $\mathcal{M}$.

Thus, we have defined a two-tier differentiated service scheme: if the observed delay D of the transaction is lower than or equal to t_+ (that has been calculated to be consistent with the server-advertised parameters), the outcome quality $q = q_+$ is considered to belong to the *high QoS* tier and $\psi = \psi_+$. On the other hand, if $t_+ < D < t_-$, the outcome quality $q = q_-$ is considered to belong to the *low QoS* tier and $\psi = \psi_-$. Finally, if $D > t_-$ the transaction is considered to have failed, and the server is penalized.

In order to apply the model in Section 2.1 (and thus obtain the optimum contract prices ψ_- and ψ_+), peers need

[5] We choose TCP only for analytical convenience. Extending the model to use other transport protocols, such as RTP [28] or TFRC [17], would require more explicit consideration of the channel loss probability, but is otherwise a simple matter. The performance of TCP for media streaming, however, has been analytically studied and found to be good if the achievable throughput is roughly twice the media bit-rate [35].

D	Chunk delivery delay
t_{RTT}	Layer 3 RTT
S_r	Request message size
T_c	Client to server throughput
S_c	Chunk Size
T_s	Server to client throughput
t_P	Request processing time at the server
N_s	Number of peers amongst which the server upload is shared

Table 2. Transaction delay components

S_p	Packet size at layer 3
p_L	Packet loss probability at layer 3
t_{RTO}	Retransmission timeout

Table 3. TCP Channel model parameters

to estimate p_+, p_-, ϕ_+ and ϕ_- as a function of the prevailing network conditions and the desired deadline for the transaction, t_+.[6]

To accomplish this, we require a model linking the transaction-level delay (the process that defines the QoS received by clients) with the packet-level delay (the varying network conditions). Thus, we seek to express the distribution the transaction delay in terms of the relevant parameters of the round trip packet delay and the model (4) (Figure 1). To do this, we note that the RTT delays can be modeled using a shifted gamma distribution [24, 4], so that the two shape parameters of the gamma distribution (θ and k), along with a shift parameter Δ can be fit from data on the prevailing characteristics of the RTT delay on the link. We can thus express the RTT delay PDF as:

$$f(x) = \frac{1}{\theta^k \Gamma(k)}(x-\Delta)^{k-1} e^{-\frac{x-\Delta}{\theta}} \tag{5}$$

By transforming f in (5) according to (4) and introducing a simple model for TCP throughput as parametrized by RTT [25], we find that the transaction-level delay distribution g is (see Appendix B):

$$g(y) = \frac{1}{(\xi_1\theta)^k \Gamma(k)}(y - t_P - \xi_2 - \xi_1\Delta)^{k-1} e^{-\frac{y-t_P-\xi_2-\xi_1\Delta}{\xi_1\theta}} \tag{6}$$

Where:

$$\xi_1 = 1 + \sqrt{\frac{2p_L}{3}}\frac{S_r + N_s^* S_c}{S_p}$$

$$\xi_2 = t_{RTO}(3\sqrt{\frac{3p_L}{8}})p_L(1+32p_L^2)\frac{S_r + N_s^* S_c}{S_p}$$

[6] Peers define t_- in a completely private fashion.

(see Tables 2 and 3 for nomenclature). The derivation of this result can be found in Appendix B.

For the rest of the parameters in the model, we define the following mapping between the observable variables and ϕ and q:[7]

$$\phi = \kappa_1 \left(\frac{1}{N_s t_P}\right)^{\epsilon} \quad q = \kappa_2 \frac{1}{D^{\eta}} \tag{7}$$

where $\kappa_1 \in \mathbb{R}_{\geq 0}, \kappa_2 \in \mathbb{R}_{\geq 0}, \eta \in (0,1), \gamma \in (0,1), \epsilon \in (0,1)$. For ϕ_+ and q_+, the server is assumed to have operated according to its advertised parameters, and:

$$\phi_+ = \kappa_1 \left(\frac{1}{N_s^+ t_P^+}\right)^{\epsilon} \quad q_+ = \kappa_2 \frac{1}{D_+^{\eta}}$$

To calculate t_+, the client queries $\mathcal{M}$ for the server-advertised t_P^+ and N_s^+ and defines a target p_+ that represents the tolerance that the client is willing to give to the server ($1 - p_+$ is the probability that the transaction will conclude after t_+ even though the server upholds its advertised t_P^+ and N_s^+, and thus also measures the risk involved for the server). From (5), and referring to Figure 2, we see that t_+ is found by solving:

$$\Gamma(k)p_+ = \gamma\left(k, \frac{t_+ - t_P^+ - \xi_2 - \xi_1\Delta}{\theta\xi_1}\right) \tag{8}$$

where $\gamma(k,x)$ is the *incomplete gamma function* (See Appendix B).

We now model the two classes of service related to the effort levels ϕ_+ and ϕ_-. We refer the reader to Figure 2 for a graphical representation of the time intervals and probabilities related to the following discussion.

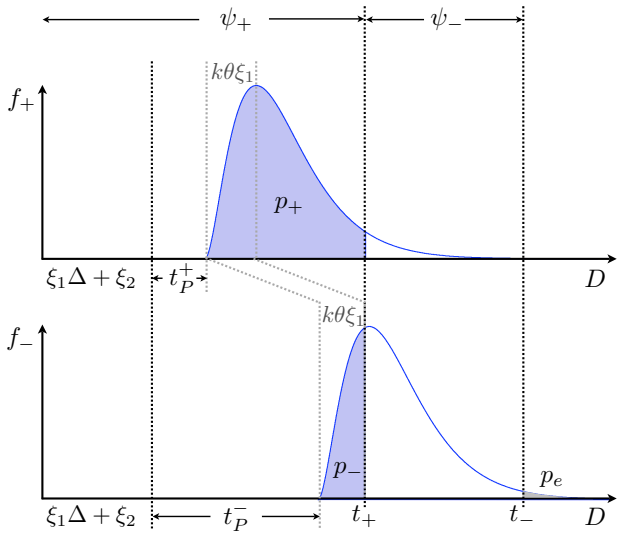

Figure 2. Determination of t_P^+ and t_P^-

The transaction delay distribution f_+ associated with high-QoS can be obtained from (5) by setting $t_P = t_P^+$. In order to obtain f_-, we propose that if the server fails to uphold t_P^+ and N_s^+, it will choose an alternative effort level defined by t_P^- and N_s^- in which it attempts to put as little effort as possible, while trying to maintain high payment ψ_+. For the purpose of this paper, we assume that the server reveals N_s truthfully in all circumstances, and thus $N_s^- = N_s^+$. Thus, for ϕ_-, the server will decrease its effort by increasing its processing delay so that $t_P = t_P^- < t_P^+$, and it will calculate t_P^- so that the expected value of f_- is equal to t_+, thus selecting the lowest effort that, in expectation, still leads to a high paying outcome. This means that the client chooses the t_P^- that makes the following expression hold (see Figure 2):

$$\xi_1 \Delta + \xi_2 + t_P^- + k\theta\xi_1 = t_+ \tag{9}$$

Clearly, the service level defined by (t_P^-, N_s^+) is significantly worse than that one implied by (t_P^+, N_s^+), as normally the expected transaction resolution time if the server upholds t_P^+ is smaller than t_+. Once that t_P^- is defined, we calculate p_- by setting $t_P = t_P^-$ in (6) and finding the integral up to t_+. Equivalently:

$$p_- = \frac{\gamma\left(k, \frac{t_+ - t_P^- - \xi_2 - \xi_1\Delta}{\theta\xi_1}\right)}{\Gamma(k)} = \frac{\gamma(k,k)}{\Gamma(k)}$$

For ϕ_- and q_-, since the client assumes that the server has actually tried to deliver as low quality of service as it can get away with, we have that:

$$\phi_- = \kappa_1 \left(\frac{1}{N_s^+ t_P^-}\right)^{\epsilon} \quad q_- = \kappa_2 \frac{1}{D_-^{\eta}}$$

Clearly, it is in the best interest of the server to minimize its probability of being paid ψ_-. Thus, the server will only accept contracts for which it can deliver high values for p_+, which in turn gives it an incentive to truthfully reveal the effort level that it is willing and able to maintain.

Depending on the the value that the client assigns to t_-, there might be a possibility that the transaction could exceed t_-, with no malicious intent by the server. If this probability, p_e (see Figure 2) presents an undesirably high risk, the server can choose to reject the contract and refuse service. the same considerations apply if p_+ is too low.

In Figure 3 we see a graphical representation of expressions (2) and (3) in the context of the model presented in this section. Figure 3 shows the payments ψ_+ that a client would offer to a server as a function of its advertised effort, for an scenario with parameters as shown in Table 4. Thus,

[7] κ_1, κ_2, γ, ϵ and η are external parameters that can be used to fit these functions to experimental measurements. For the rest of this paper, we consider them external parameters that are given.

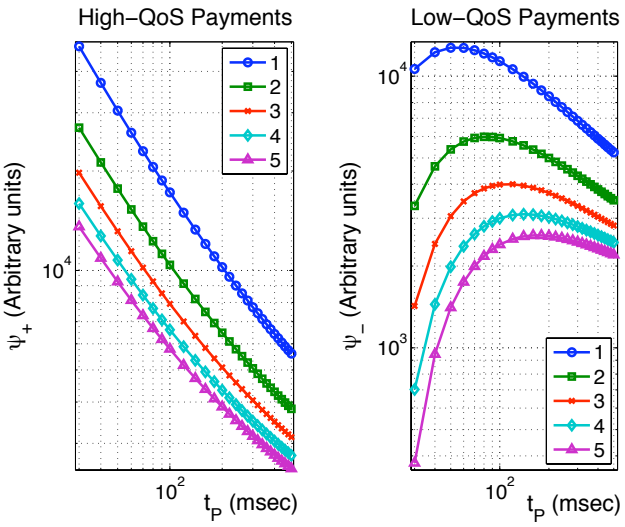

Figure 3. ψ_+ and ψ_- for the simulation run of Table 4 (the legend in both graphs shows different values of N_s, the number of clients that the server is serving simultaneously).

Mean t_{RTT}	110 msec
k	3
θ	3
S_r	200 bytes
S_c	64 kbyte
S_p	1500 bytes
p_L	10^{-3}
t_{RTO}	100 msec

Table 4. Simulation parameters

a server peer that advertises in $\mathcal{M}$ that it is able to commit to an effort level corresponding to $N_s^+ = 5$ and $t_P^+ = 30$ msec. can expect to be offered by a prospective client a contract where $\psi_+ \approx 13500$ and $\psi_- \approx 370$ in $\mathcal{M}$ currency units. Thus, the contract would be drafted with these two payment values. Following the outcome of the transaction, the server will receive ψ_+ if the transaction is successful with high QoS, ψ_- if the transaction is successful with low QoS, and nothing if the transaction is unsuccessful.

The nonlinear behavior of ψ_- near the origin is a product of the sensitivity of our definition for ϕ to very small decreases of t_P in the vicinity of zero, as it is clear from (7).

3.1. Contract Enforcement

The model in Section 2.1 assumes that the contract is enforceable, as it often the case in economics (usually through

	Server Cooperate	Server Defect
Client Cooperate	(U_c^+, U_r)	$(-\tau, 0)$
Client Defect	$(U_c^d, -\phi_+)$	$(-\tau, 0)$

Table 5. Repeated game, normal form

reliance on a legal framework). However, in the case of peer-to-peer overlay networks, such enforcement is difficult to provide because peers can lie about transaction outcomes, and this can be exploited to execute denial of service attacks against particular peers, greatly increasing the policing costs associated with each transaction. The application of well-known techniques of community enforcement [18] is complicated by the fact that peer interactions are unobservable beyond the relevant actors for the transaction, and thus open to slander attacks. Instead, we propose a scheme based on the creation of *social capital* [7] by directly controlling the interaction graph of the peers. We model this peer-based enforcement as reciprocity-based trigger strategies on a repeated game.

If peers follow the algorithm in Section 2.1 for the creation of contracts, but these contracts are not enforceable and thus are susceptible to *moral hazard*[8], we can model this as a repeated game where each stage game has the normal form shown in Table 5, where U_c^d, the *defection utility*, corresponds to:

$$U_c^d = p_+ q_+ + (1 - p_+) q_-$$

and τ is a measure of time lost to the client waiting for a server response before timeout.

We model each transaction as a stage in a repeated game. We assume peers to have two strategies: adherence to the protocol, or selfish utility maximization. Following the usual nomenclature for the *Prisoner's Dilemma*, we call adherence to the protocol *cooperation* and deviation from it (failure to uphold contractual obligations) as *defection*.

If $\mathcal{M}$ ensures that peers maintain a relatively stable relationship, so that the probability of peers interacting with each other remains under its control, it is possible to ensure that it is nonprofitable for the peers to fail to uphold their contracts.

To guide the Nash equilibrium of this repeated game, $\mathcal{M}$ explicitly modifies the information that it provides to the peers in order to ensure a minimum probability of repeated interaction δ (usually called the *discount parameter* in game theory). By forcing the δ between peers to the value where deviation from the protocol becomes unprofitable, we can ensure that only irrational peers will deviate.

[8]In situations with moral hazard, agents modify their behavior toward risky activities depending on whether the consequences of these risks apply to themselves or to other agents.

We explore two possible trigger strategies for contract enforcement: *Grim Trigger* and *Tit-for-Tat* [3].

3.1.1 Grim Trigger Enforcement

As usual, in this strategy a given peer i follows the protocol towards another peer j as long as j follows the protocol towards it. If j deviates, however, i retaliates by following a strategy of continuous defections from then onwards. thus, any protocol deviation receives the harshest punishment possible, and there is no possibility for "forgiveness" and re-introduction of a possible cooperative regime. In this case, $\mathcal{M}$ must ensure that (see Appendix C):

$$\delta_G \geq \frac{p_+\psi_+ + (1-p_+)\psi_-}{p_+ q_+^\beta + (1-p_+) q_-^\beta + \tau}$$

3.1.2 Tit-for-Tat Enforcement

In this strategy a given peer i at a stage t applies the strategy that its partner j applied at a stage $t-1$. In this case, we have that $\mathcal{M}$ must ensure that (see Appendix D):

$$\delta_T \geq \frac{p_+\psi_+ + (1-p_+)\psi_-}{p_+(q_+^\beta - \psi_+) + (1-p_+)(q_-^\beta - \psi_-) + \tau}$$

4. Related Work

Most of the work focusing in the strategic elements of QoS on overlay networks assume that clients are able to equate server effort with the experienced QoS. Thus, although the strategic quality of QoS is widely recognized, the fact that it might be impossible to measure it directly has not received nearly as much attention.

There have been several attempts to model contracts in the context of peer-to-peer systems. In [16], the authors propose a system where peers agree to a contract when joining the network. This contract specifies the amount of resources that they will be required to contribute to the system in order to gain access to its services. The authors cast the optimal contract calculation as a dynamic programming problem that seeks to assign to incoming peers a set of obligatory resource donations that is mutually beneficial for the peer and the network (both their utilities increase). As a particular example of this, in [19] the authors consider a peer-to-peer file sharing system where each download must be offset by uploading it up to N times within a maximum time interval T.

With respect to the modeling of hidden action in networking scenarios, there have been several attempts to use the principal-agent model in routing and peer-t-peer systems. In [12, 11], the authors propose a system where routing is analyzed using a principal-agent model. The endpoints of the flow (the sender and the receiver) are modeled as the principal, and the forwarding nodes as agents.

The authors derive expressions for optimal contracts that can be used to elicit high effort on the part of the agents, while minimizing investment by the principal. Furthermore, the authors find that the usefulness of information regarding outcomes at intermediate stages in the routing process is contingent on the network topology, and that contracts can be implemented both directly (between the principal and each agent) or recursively (between every node and its downstream partner).

In [23], the author analyzes hidden action from the standpoints of market theory and communitarian resource allocation. Based on qualitative arguments, he posits that by constraining the overlay network topology to consist of fully connected cliques interconnected through a smaller number of stable, long lived channels. Thus, the author posits, peers would be forced to have longer lived interactions with a smaller number of peers, making reciprocative strategies more effective.

5. Conclusions

In this paper we develop a model for the design and verification of QoS contracts in peer-to-peer overlays implemented over best-effort networks, and apply it to a specific case: a mesh-based, pull-oriented streaming system with low delay requirements. We treat each transfer of a delay-sensitive chunk as a *hidden action* situation where server peers can force negative externalities upon client peers, and we propose a contract drafting procedure that allows peers to deploy incentives to counter hidden action. This is an early step in the analytic treatment of QoS as a strategic peer behavior, and towards a general theory of service level agreements among strategic peers. We show that the construction of optimal contracts through the use of the *principal-agent* model can be used to ensure predictable QoS behavior, even with utility-maximizing rational peers whose service differentiation processes are unobservable. We do this by requiring the server peer to accept part of the externality that would have been exclusively endured by the client.

Additionally to the solved general model, we also present its application to a chunk transfer scenario where transaction delay is the main QoS indicator. In this case, we show how to estimate the model parameters from truthfully revealed values and the analysis of observable network conditions.

Finally, we study the enforcement of these contracts as a repeated game, and show that a centralized entity can define the resulting Nash equilibrium to be a cooperative one by specifying the minimum probability of pairwise repeated interaction (discount parameter).

References

[1] E. Adar and B. Huberman. Free riding on gnutella. Technical report, Xerox PARC, August 2000.

[2] K. G. Anagnostakis and M. B. Greenwald. Exchange-based incentive mechanisms for peer-to-peer file sharing. In *ICDCS '04: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, pages 524–533, Washington, DC, USA, 2004. IEEE Computer Society.

[3] R. Axelrod and W. D. Hamilton. The evolution of cooperation. *Science*, 211, March 1981.

[4] J.-C. Bolot. End-to-end packet delay and loss behavior in the internet. In *SIGCOMM '93: Conference proceedings on Communications architectures, protocols and applications*, pages 289–298, New York, NY, USA, 1993. ACM.

[5] S. C. Choi and R. Wette. Maximum likelihood estimation of the parameters of the gamma distribution and their bias. *Technometrics*, 11(4):683–690, November 1969.

[6] B. Chun, Y. Fu, and A. Vahdat. Bootstrapping a distributed computational economy. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.

[7] J. S. Coleman. Social capital in the creation of human capital. *The American Journal of Sociology*, 94:S95–S120, 1988.

[8] J. R. Douceur. The sybil attack. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 251–260, London, UK, 2002. Springer-Verlag.

[9] J. Feigenbaum and S. Shenker. Distributed algorithmic mechanism design: recent results and future directions. In *DIALM '02: Proceedings of the 6th international workshop on Discrete algorithms and methods for mobile computing and communications*, pages 1–13, New York, NY, USA, 2002. ACM Press.

[10] M. Feldman and J. Chuang. Overcoming free-riding behavior in peer-to-peer systems. *SIGecom Exch.*, 5(4):41–50, 2005.

[11] M. Feldman, J. Chuang, I. Stoica, and S. Shenker. Hidden-action in multi-hop routing. In *EC '05: Proceedings of the 6th ACM conference on Electronic commerce*, pages 117–126. ACM, 2005.

[12] M. Feldman, J. Chuang, I. Stoica, and S. Shenker. Hidden-action in network routing. *Selected Areas in Communications, IEEE Journal on*, 25(6):1161–1172, August 2007.

[13] M. Feldman, K. Lai, I. Stoica, and J. Chuang. Robust incentive techniques for peer-to-peer networks. In *EC '04: Proceedings of the 5th ACM conference on Electronic commerce*, pages 102–111, New York, NY, USA, 2004. ACM.

[14] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica. Free-riding and whitewashing in peer-to-peer systems. In *PINS '04: Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, pages 228–236, New York, NY, USA, 2004. ACM Press.

[15] D. F. Ferguson, C. Nikolaou, J. Sairamesh, and Y. Yemini. Economic models for allocating resources in computer systems. *Market-based control: a paradigm for distributed resource allocation*, pages 156–183, 1996.

[16] D. Ghosal, B. K. Poon, and K. Kong. P2p contracts: a framework for resource and service exchange. *Future Gener. Comput. Syst.*, 21(3):333–347, 2005.

[17] M. Handley, S. Floyd, J. Padhye, and J. Widmer. Tcp friendly rate control (tfrc): Protocol specification, 2003.

[18] M. Kandori. Social norms and community enforcement. *Review of Economic Studies*, 59(1):63–80, January 1992.

[19] B. Khorshadi, X. Liu, and D. Ghosal. Determining the peer resource contributions in a p2p contract. *Hot Topics in Peer-to-Peer Systems, 2005. HOT-P2P 2005. Second International Workshop on*, pages 2–9, 21 July 2005.

[20] D. Kostić, A. Rodriguez, J. Albrecht, and A. Vahdat. Bullet: high bandwidth data dissemination using an overlay mesh. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 282–297, New York, NY, USA, 2003. ACM.

[21] Z. Li and P. Mohapatra. Qron: Qos-aware routing in overlay networks. *IEEE Journal on Selected Areas in Communications*, 22(1):29–40, January 2004.

[22] J. A. Mirrlees. The theory of moral hazard and unobservable behaviour: Part i. *Review of Economic Studies*, 66(1):3–21, January 1999.

[23] T. Moore. Countering hidden action attacks on networked systems. *Fourth Workshop on the Economics and Information Security*, June 2005.

[24] A. Mukherjee. On the dynamics and significance of low frequency components of internet load. *Internetworking: Research and Experience*, 5:163–205, December 1994.

[25] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling tcp throughput: a simple model and its empirical validation. *SIGCOMM Comput. Commun. Rev.*, 28(4):303–314, 1998.

[26] F. Pianese, J. Keller, and E. W. Biersack. Pulse, a flexible p2p live streaming system. *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–6, April 2006.

[27] S. Sanghavi and B. Hajek. A new mechanism for the free-rider problem. In *P2PECON '05: Proceeding of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems*, pages 122–127, New York, NY, USA, 2005. ACM Press.

[28] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. Rtp: A transport protocol for real-time applications, 2003.

[29] L. Subramanian, I. Stoica, H. Balakrishnan, and R. Katz. Overqos: offering internet qos using overlays. In *ACM SIGCOMM CCR*, volume 33, pages 11–16, January 2003.

[30] K. Tamilman, V. Pai, and A. Mohr. Swift: A system with incentives for trading. In *Proceedings of Second Workshop of Economics in Peer-to-Peer Systems*, 2004.

[31] D. A. Turner and K. W. Ross. A lightweight currency paradigm for the P2P resource market. In *7th International Conference on Electronic Commerce Research*, June 2004.

[32] V. Vishnumurthy, S. Chandrakumar, and E. G. Sirer. Karma: A secure economic framework for p2p resource sharing. In *Proc. of Workshop on the Economics of Peer-to-Peer Systems*, 2003.

[33] B. D. Vleeschauwer, F. D. Turck, B. Dhoedt, and P. Demeester. On the construction of qos enabled overlay networks. *Quality of Service in the Emerging Networking*, pages 164–173, September 2004.

[34] C. A. Waldspurger, T. Hogg, B. A. Huberman, J. O. Kephart, and W. S. Stornetta. Spawn: A distributed computational economy. *IEEE Trans. Software Eng.*, 18(2):103–117, 1992.

[35] B. Wang, J. Kurose, P. Shenoy, and D. Towsley. Multimedia streaming via tcp: an analytic performance study. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 908–915, New York, NY, USA, 2004. ACM.

[36] W. Wang and B. Li. Market-driven bandwidth allocation in selfish overlay networks. In *INFOCOM 2005, 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 2578–2589. IEEE, 2005.

[37] X. Zhang, J. Liu, B. Li, and Y.-S. Yum. Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming. *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, 3:2102–2111 vol. 3, 13-17 March 2005.

[38] Z. Zhang, S. Chen, and M. Yoon. March: A distributed incentive scheme for peer-to-peer networks. *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 1091–1099, May 2007.

A Optimal contract prices for a Hidden Action Model

The optimization problem (1) is equivalent to:

$$\begin{aligned} \text{Maximize:} \quad & p_+(q_+^\beta - \psi_+) + (1-p_+)(q_-^\beta - \psi_-) \\ \text{Subject to:} \quad & p_+\psi_+^\alpha + (1-p_+)\psi_-^\alpha - \phi_+ \geq U_r \\ \text{And:} \quad & p_+\psi_+^\alpha + (1-p_+)\psi_-^\alpha - \phi_+ \\ & \geq p_-\psi_+^\alpha + (1-p_-)\psi_-^\alpha - \phi_- \end{aligned}$$

We construct the following Lagrangian:

$$\begin{aligned} L(\psi_+, \psi_-) = \quad & p_+(q_+^\beta - \psi_+) + (1-p_+)(q_-^\beta - \psi_-) \\ & + \lambda(p_+\psi_+^\alpha + (1-p_+)\psi_-^\alpha - \phi_+ - U_r) \\ & + \mu((p_+ - p_-)(\psi_+^\alpha - \psi_-^\alpha) - \phi_+ + \phi_-) \end{aligned}$$

The critical point requirement $\nabla L = \mathbf{0}$ yields the following first-order conditions:

$$\alpha\psi_+^{(\alpha-1)}(p_+\lambda + \mu(p_+ - p_-)) = p_+$$

$$\alpha\psi_-^{(\alpha-1)}((1-p_+)\lambda + \mu(p_+ - p_-)) = (1-p_+)$$

$$p_+\psi_+^\alpha + (1-p_+)\psi_-^\alpha = U_r + \phi_+ \tag{10}$$

$$(p_+ - p_-)(\psi_+^\alpha - \psi_-^\alpha) = \phi_+ - \phi_- \tag{11}$$

By obtaining ψ_+^α from (10) and substituting on (11), we find that:

$$U_r + \phi_+ - \psi_-^\alpha = \frac{p_+(\phi_+ - \phi_-)}{p_+ - p_-}$$

Thus:

$$\psi_- = \left(U_r + \frac{p_+\phi_- - p_-\phi_+}{p_+ - p_-}\right)^{\frac{1}{\alpha}} \quad (12)$$

By substituting (12) on (11) we find that:

$$\psi_+ = \left(U_r + \frac{(1-p_-)\phi_+ - (1-p_+)\phi_-}{p_+ - p_-}\right)^{\frac{1}{\alpha}}$$

B Transaction time distribution model

We require a model linking the strategic behavior of the server and the expected behavior of the network. If we assume that t_{RTT} is constant for the entirety of the transaction, and identical TCP stack configurations in the client and the server, and that TCP sharing is approximately fair amongst flows in the server, we have that (From [25], using the nomenclature in Tables 2 and 3):

$$T_c = T_s = \frac{S_p}{t_{RTT}\sqrt{\frac{2p_L}{3}} + t_{RTO}(3\sqrt{\frac{3p_L}{8}})p_L(1+32p_L^2)}$$

Thus, substituting in (4) we have that:

$$D = \xi_1 t_{RTT} + t_P + \xi_2$$

Where

$$\xi_1 = 1 + \sqrt{\frac{2p_L}{3}}\frac{S_r + S_c}{S_p}$$

$$\xi_2 = t_{RTO}(3\sqrt{\frac{3p_L}{8}})p_L(1+32p_L^2)\frac{S_r + N_s S_c}{S_p}$$

It is well known that if a random variable X has a probability density function $f(x)$, then $Y = h(x)$ as the following PDF:

$$g(y) = f(h^{-1}(y))\left|\frac{\partial(h^{-1}(y))}{\partial y}\right|$$

Thus, if we set $h(x) = \xi_1 t_{RTT} + x + \xi_2$ we have the following result for the distribution of total transaction delays:

$$g(y) = \frac{1}{\Gamma(k)\theta\xi_1}\left(\frac{y - t_P - \xi_2 - \xi_1\Delta}{\theta\xi_1}\right)^{k-1} e^{-\left(\frac{y-t_P-\xi_2-\xi_1\Delta}{\theta\xi_1}\right)}$$

The shape parameters Δ, θ and k can then be obtained from RTT measurements using maximum likelihood estimation [5]. For the CDF we have that $P(y < t) = \int_0^t g(y)dy$ and we have:

$$P(y < t) = \frac{\gamma\left(k, \frac{t-t_P-\xi_2-\xi_1\Delta}{\theta\xi_1}\right)}{\Gamma(k)}$$

where $\gamma(k, x)$ is the *incomplete gamma function*.

C Contract Enforcement: Grim Trigger

From the game normal form, it is clear that only the client has an incentive to defect. Without loss of generality, we consider the case where the client defects on the first round, and the server retaliates by defecting from then on. If we take into account the discount parameter δ defined by $\mathcal{M}$ as the probability that i and j will meet again to play the game, we have the following expected utility for the client:

$$U^g = U_c^d - \frac{\delta}{1-\delta}\tau$$

By considering the requirement that $U^g < U_c^+$, we have that:

$$p_+ q_+^\beta(1-\delta) + (1-p_+)q_-^\beta(1-\delta) - \delta\tau \leq p_+(q_+^\beta - \psi_+) + (1-p_+)(q_-^\beta - \psi_-)$$

Simplifying, we find that:

$$\delta(p_+ q_+^\beta + (1-p_+)q_-^\beta + \tau) \geq p_+\psi_+ + (1-p_+)\psi_-$$

And thus:

$$\delta \geq \frac{p_+\psi_+ + (1-p_+)\psi_-}{p_+ q_+^\beta + (1-p_+)q_-^\beta + \tau}$$

D Contract Enforcement: Tit-for-Tat

In this case, the first defection from the client starts a series of alternating cooperation-defection episodes. For the expected utility for the client we have:

$$U^t = U_c^d - \delta\tau + \delta^2 U_c^d - \delta^3\tau + \ldots = (1 + \delta^2 + \delta^4 + \ldots)(U_c^d - \delta\tau)$$

Again, by considering the requirement that $U^t < U_c^+$, we have that:

$$\frac{1}{1-\delta}U_c^+ \geq \frac{1}{1-\delta^2}(U^t - \delta\tau)$$

Simplifying, we find that:

$$\delta \geq \frac{U^t - U_c^+}{U_c^+ + \tau}$$

And thus:

$$\delta \geq \frac{p_+\psi_+ + (1-p_+)\psi_-}{p_+(q_+^\beta - \psi_+) + (1-p_+)(q_-^\beta - \psi_-) + \tau}$$