A Markov Model of Server to Client IP traffic in First Person Shooter Games

Philip A. Branch, Antonio L. Cricenti, Grenville J. Armitage
Centre for Advanced Internet Architectures
Swinburne University of Technology
Melbourne, Australia

***Abstract*—Modeling traffic generated by Internet based multiplayer computer games has attracted a great deal of attention in the past few years. In part this has been driven by a need to simulate the network impact of highly interactive online games such as the first person shooter (FPS). Packet size distributions and autocorrelation models are important elements in the creation of realistic traffic generators for network simulators. In this paper we present a simple technique for constructing Markov chains that model autocorrelated packet length for N-player FPS games based on traffic traces of of 2-player games. This enables us to synthesize the time sequence of the length of server to client traffic as well as its probability distribution. We illustrate the likely generality of our approach using data from seven FPS games that have been popular over the past nine years: Half-Life, Half-Life Counterstrike, Half-Life 2, Half-Life 2 Counterstrike, Quake III Arena, Quake 4 and Wolfenstein Enemy Territory.**

***Index Terms*—Online Games, Traffic Modeling, Simulation**

I. Introduction

Modeling traffic generated by Internet based multiplayer computer games has attracted a great deal of attention in the past few years [1-14]. Highly interactive genres such as the First Person Shooter (FPS) are of particular interest to network engineers. Like voice over IP (VoIP) and other interactive conference-style applications, FPS games are generally intolerant of packet loss, jitter and high latency. FPS games commonly use User Datagram Protocol (UDP) over IP for transport and so do not adjust packet rates in response to network congestion. Finally, FPS games are typically based on a client-server model for network traffic, with thousands or tens of thousands of FPS servers active on the Internet at any given time [15]. This has motivated research community interest in predicting and simulating the traffic load imposed on network links by multiplayer FPS games.

Two questions are of particular interest - how traffic generated by FPS games increases as the number of players increases, and how this traffic affects, and is affected by, other traffic sharing the network. Since it is usually impractical to build and measure a full-size network, the second question is typically answered through simulation using statistical models created from the answers to the first question.

Traffic in the client to server direction usually consists of small IP packets whose size distribution is independent of the number of players on a given server. However, traffic in the server to client direction usually shows distinct variation as the number of players increases [15]. Published work to date has typically involved empirical studies of FPS games in small test beds with up to 8 to 10 players. Traffic models have been created that match the statistical packet size distributions for each *N*-player game, for *N* = 2, 3, and so on. Yet some public FPS game servers may be configured to allow up to 60 players [15]. Obtaining empirical data in controlled circumstances for games with such large numbers of players is challenging. There is a need for techniques that allow extrapolation of statistical characteristics from games with small numbers of players to games with much larger numbers of players.

In this paper we propose and illustrate a technique for extrapolating *N*-player traffic statistics from empirically measured traffic of 2-player FPS games. In particular we focus on predicting the distribution and autocorrelation of packet sizes in the server to client direction. This allows small-scale empirical measurements to be applied to larger scale simulations of FPS traffic acting on an IP network. The simulated traffic captures the time sequence of the length of server to client traffic as well as its probability distribution. We illustrate the likely generality of our approach using data from seven FPS games released between 1998 and 2006: Half-Life, Half-Life Counterstrike, Quake III Arena, Quake 4, Wolfenstein Enemy Territory, Half-Life 2 and Half-Life 2 Counterstrike.

Although work on modeling of game traffic has been carried out for some time now, much of it has been empirical. The contribution of this paper is to show how empirical measurements can be extended to predict the behavior of game traffic for larger numbers of players.

The paper is structured as follows. Section II reviews the basic network architecture and traffic patterns of modern FPS games. Section III introduces our proposed method of predicting *N*-player distributions from games with smaller numbers of players. Section IV introduces a number of example distributions from the seven FPS games we tested. Section V describes a Markov chain model that captures the autocorrelated nature of FPS game traffic. Section VII discusses future research and concludes the paper.

II. First Person Shooter Games

A. Game State Updates

Multiplayer online games have an underlying requirement that game-state information is shared amongst all players in near real-time. Each game client acts as an interface between the local human player and the virtual game-world within which the player interacts with other players. In principle clients might be designed to communicate directly with each other in a peer-to-peer fashion. In practice, most FPS games use a client-server model (including the seven examples presented in this paper). Every client's actions are sent in short messages to the server, and every client is regularly updated with the actions taken by other players and their consequences. The server implements the game-world's state machine, regulating client actions in order to maintain the game's internal rules and minimize opportunities for cheating.

A typical FPS game involves an ISP or game enthusiast hosting a game server on the Internet, and players joining the game using client software running on a home PC or IP-enabled game console. Games can also be run on a private, local IP network – commonly referred to as 'LAN parties'. The game client updates and renders the game's virtual world on the client's screen based on messages received regularly from the game server. User inputs to the game client (actions such as walking, exploring or shooting weapons) are passed to the game server to be verified and propagated to other players.

Game-state updates must occur in a timely manner, with minimal bias towards any particular player. In FPS games, timeliness is achieved by sending a unicast IP packet to each client every Y milliseconds (ms). Y is typically in the range of 30 to 60ms. To minimize bias, update packets to different clients are sent in back-to-back bursts [15]. Each client receives an update packet every Y ms regardless of how much in-game activity is occurring. The choice of Y for a given game depends on the available network capacity (longer Y for lower bandwidth demand) versus player expectations of smooth interactivity (shorter Y for more frequent updates).

Clients send their own updates to the game server at less precisely defined intervals, often influenced by the client's processor speed, graphics card settings and player activity. Typical intervals vary from 10ms to 40ms [15].

B. *Phases of Game-play and Game Traffic*

In most FPS games there are three different phases of interaction between client and server that impact on network traffic.

- A client connects to the server, and receives data from the server to update the client's local virtual world information (map definitions, avatar 'skins', etc).
- The client is connected to the server and the game is in progress (players run around the virtual world, shooting or otherwise interacting with each other).
- The client is connected to the server, and the game has been paused as the server changes maps or restarts a previous map (after a player wins the previous 'round')

Tight control over network jitter and packet loss is only essential during active game-play. During periods of player inactivity (initial client connection and server changing maps) the network can exhibit fluctuating latency, jitter and packet loss without affecting the player's game experience.

C. *Traffic Compression*

To maximize playability over a wide range of network conditions and consumer access technologies modern FPS games actively compress the data sent over the network. For example, an examination of the Quake III Arena source code [16] shows that every 50 milliseconds the server sends an update message to each client that contains the current game time, a status bar update and an UPDATE ENTITY message. The UPDATE ENTITY message is heavily compressed using delta compression and only contains information about players and objects currently visible within the game world to the receiving client. With delta compression if there is no change to a visible entity, no additional information is transmitted. If the entity has changed its state significantly, for example, if it has moved or is shooting at another player, much more information is sent.

The number of visible entities depends on the number of players and the virtual world's layout (the 'map'). For example, maps with many walls and corridors will result in less visibility between players (and less information per update packet on average) than maps with wide-open areas. Likewise, a map containing many players will experience many more player-player interactions (per unit time) than a map with few players scattered around the virtual game world.

As regards client to server traffic, clients generate events describing a single player's activity. A typical human can trigger only a limited number of events in any given 10ms to 40ms window. Consequently packets from client to server are typically much smaller than the packets from server to client, and exhibit very limited variation in size. For example, during active game play, the client to server traffic in Quake III Arena is mostly composed of a single message - the CLIENT MOVEMENT message. This contains information as to direction and speed as well as flags for jumping and firing. No other information is transmitted. The range of values is quite small - between between 25 and 45 bytes with 90% of all packets between 28 and 38 bytes.

An important consequence of traffic compression in Internet based FPS games is that it is difficult to predict anything other than the simplest characteristics of game traffic from a line-by-line analysis of the source code. Because the traffic transmitted depends so strongly on the random behavior of the players, and the nature of the game map, a stochastic approach to modeling game traffic is often the most useful way of understanding game traffic. We adopt this approach in this paper.

III. Modeling Server to Client Traffic

In this section we show how simple assumptions as to the nature of player behavior and game design can be used to extrapolate from games with small numbers of players to games with larger numbers of players. We have presented a technique that allowed packet lengths for N-player games to be successfully predicted based on data from two and three player games [2]. We now show how a Markov chain model of server to client game traffic can be constructed in a similar

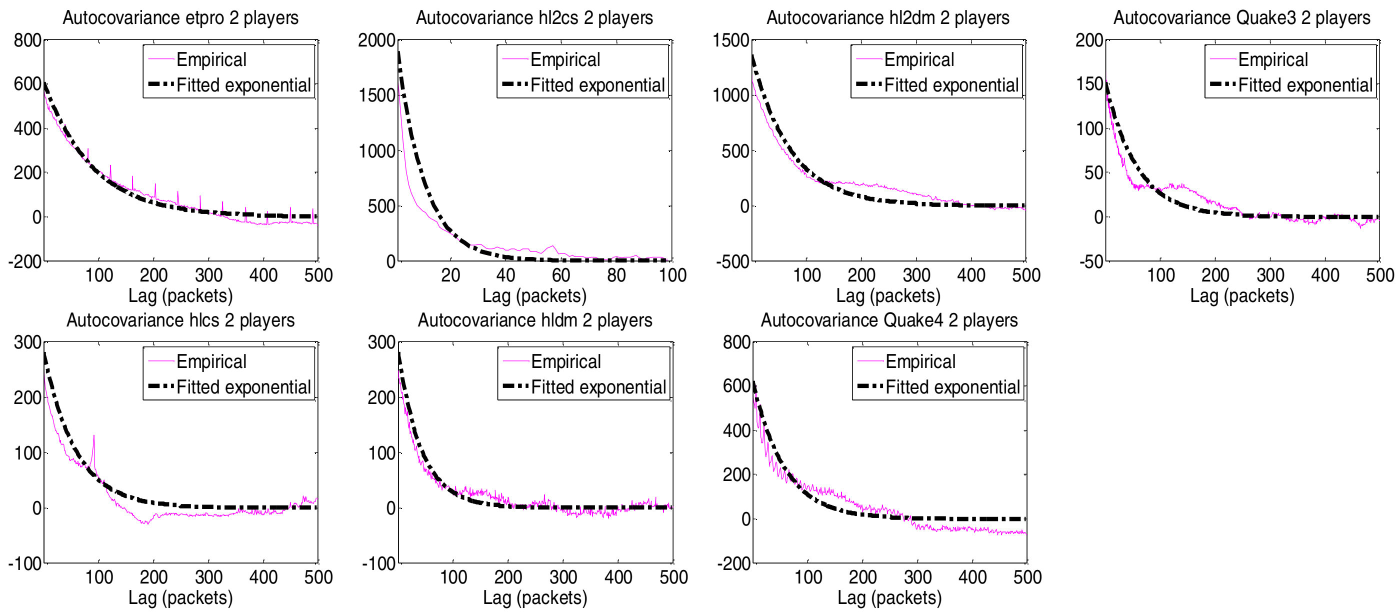

Figure 1. Two player game autocovariance plots for ETPRO, HL2CS, HL2DM, Quake3, HLCS, HLDM, Quake 4

way.

As noted before, tight control on jitter and delay is really only necessary during active game play. Consequently, we simplify our modeling by focusing on traffic patterns that exist during active game play. We further focus on server to client packet size distributions, as client to server packet length distributions are usually quite simple [15].

If we make a number of simplifying assumptions we can develop a technique for constructing Markov chain models of server to client packet lengths based on the traffic patterns of small player games. To do this we make the following simplifying assumptions:

- Individual players are likely to be involved in the same number of interactions with other players and objects regardless of the number of players.
- Players have similar behavior. They may not be of similar ability but will engage in similar activities in much the same way as each other.

Essentially these assumptions propose that the random variable describing the packet length of an N-player game can be constructed through adding together the random variables describing the packet length of smaller player games. Because we assume that individual game play does not change as the number of players increases and that the players have similar behavior, we propose that, for example, the random variable describing the length of packet payloads generated by a 5-player game can be constructed by adding the two random variables that describe server to client traffic of a 2-player game and a 3-player game. Of course these are simplifying assumptions that only approximate the true nature of FPS games. Nevertheless, by making them we can develop a simple technique for predicting Markov chains that describe games with larger number of players.

In [2] we showed this approach is effective in predicting the long term distribution of server to client packet length. We now show how these assumptions can be used to predict the time correlated nature of game traffic.

IV. Markov Chain Model of FPS Game Traffic

As well as understanding the long-term distribution of server to client packet lengths, a complete simulation model must also capture the way in which packet lengths change from one packet to the next. Some simulation models of FPS traffic derived from empirical data have implicitly assumed that there is no correlation between successive packets [12, 13].

In this section we attempt to understand the time-series behavior of game traffic. We begin by using our statistics to show that Markov models are suitable for modeling the time-series behavior of game traffic. We then show how a Markov chain describing the simplified behavior of a two player game can be extrapolated to predict the time series behavior of games with larger numbers of players. Finally we compare empirically derived Markov chain models with synthetically derived models.

A. *Markov Modeling of FPS Game Traffic*

We begin by showing that FPS server to client game traffic exhibits characteristics consistent with it belonging to a sub-class of Markov models, the autoregressive model of order 1 (AR(1)). AR(1) models have the form:

$$\begin{aligned} y_0 &= \mu_0, \\ y_{n+1} &= \beta y_n + Z_n, \quad (n \geq 1) \end{aligned} \tag{2}$$

where Z_n is a sequence of independent and identically distributed random variables (the "innovations") [17, 18].

AR(1) models have the useful identifying characteristic that their autocovariance function has the form:

$$\phi_{yy}(k) = \beta^k \sigma^2, \quad (\beta \leq 1) \tag{3}$$

where k is the lag between sample autocovariances, and β

and σ are constants. That is, the autocorrelation for an AR(1) process has the form of either a negative exponential ($\beta > 0$) or a damped cosine ($\beta < 0$). If our data has autocovariances of this form it is a strong indication that Markov modeling is appropriate.

We estimate the autocovariance function for each of the different games by

$$\phi_{yy}(\tau) = \frac{1}{T-\tau}\sum_{t=0}^{T-\tau} y(t)y(t+\tau) \qquad (4)$$

where T is the total number of samples (approximately 21000 in all cases). We determined the autocovariance function for all seven games with 2 to 9 players for all games except Quake 4 where we were limited to 5 games with 2 to 7 players.

Figure 1 shows the autocovariance function along with a best-fit exponential for a selection of the seven games with 2 players. In all cases (including those not shown) the autocorrelation function is a reasonably good match of the exponential distribution for lags up to about 500 packets, corresponding to game play durations of between 10 and 30 seconds. This gives us some confidence that Markov modeling is appropriate.

When the innovations have a well known, easy to generate distribution, simulation using the recursive definition in equation (2) is straight-forward. However, the approach adopted in this paper is to extrapolate from small player games to large player games without specifying that the characteristics of the games are described by any particular distribution. In so far as it is possible to do so, we would like to avoid specifying that the innovations used in modeling game traffic comes from any specific distribution. Instead we would prefer to show how to extrapolate time varying behavior from games with small numbers of players to games with larger numbers of players. We do this in the next section.

B. A Discrete State, Discrete Time Markov Process Model for Packet Length

We now model packet length using a Discrete State, Discrete Time Markov process (a Markov chain) and show how the Markov chain describing the two player game can be extrapolated to produce synthetic Markov chains describing games with N (greater than 2) players. We then compare the synthetic Markov chains with empirically derived Markov chains for games with 3 to 9 players [19, 20].

We model the contribution of each player to the server to client packet length by a simple Markov chain. We simplify our model by approximating the behavior of the server as follows. At each time interval, a client, through their playing behavior, causes the server to contribute either no data to the server to client packet or data of length L bytes. We denote the state of contributing no data by state 0 and the state of contributing length L bytes of data by state 1. The conditional probability of a client going from state 0 to 1 at any time interval is assumed to be constant and is denoted by p_{01}. We define p_{00}, p_{10} and p_{11} in a similar fashion. In this way the approximate behavior of the server's response to the actions of each client can be described by a Markov chain as shown in Figure 2. We obtain p_{00}, p_{01}, p_{10} and p_{11} empirically from the two player game. We then predict the aggregate behavior of N players. For N players the length of the packet generated by the server will be a multiple of L up to a maximum of NL. We determine the effectiveness of our method by binning the transition probabilities obtained from our empirical data into bin sizes of 0, L, $2L$,..., NL and compare the empirically obtained results with the synthetically obtained results.

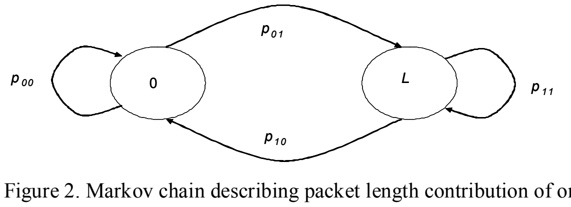

Figure 2. Markov chain describing packet length contribution of one player

The length L is obviously an important parameter. It needs to be sufficiently large to provide adequate coverage of the full range of values generated by a game of N players but must not be so large that most of the data is concentrated in one particular bin. Somewhat arbitrarily we choose this length to be the median packet length of the two-player game sample data set. This value of L provides a reasonable range of values but we make no claim as to it being the best value for L.

The next step is to assume that the behavior of N players is identical, independent and that each player's behavior can be described by the Markov chain shown in Figure 2 regardless of the number of players. We adapt the approach used in [19].

The superposition process of N identical and independent Markov processes with state space $\{0,1\}$ is a Markov Chain with state space $X = \{n, n = 0,..., N\}$ which denotes the number of players generating packets of length L at any particular timeslot.

Let $A = [a(n_1, n_2), n_1, n_2 \in X]$ be the probability transition matrix describing the transitions between the states of X. Consider two such states n_1 and n_2. Assume that at time slot k, the Markov Chain is in state n_1. The probability that in the next timeslot $k+1$ the Markov Chain will be in state n_2 is equal to the probability that l, $0 \le l \le n_1$ of the sources generating packets of length L make a transition to generating packets of length 0, and that m, $0 \le m \le N - n_2$ of the sources generating packets of length 0 make a transition to generating packets of length L and that $n_1 - l + m = n_2$.

The probability of the first event is given by the Binomial probability density function

$$\binom{n_1}{l}(1-p_{11})^l p_{11}^{\,n_1-l} \qquad (5)$$

while the probability of the second event is given by

TABLE 1.
Five player Predicted and Empirical Transition Matrices

ETPRO		HL2CS	
Predicted	Empirical	Predicted	Empirical
$\begin{pmatrix}0.71 & 0.25 & 0.04 & 0.00 & 0.00\\0.06 & 0.72 & 0.20 & 0.02 & 0.00\\0.00 & 0.11 & 0.72 & 0.15 & 0.01\\0.00 & 0.01 & 0.17 & 0.72 & 0.10\\0.00 & 0.00 & 0.03 & 0.22 & 0.70\end{pmatrix}$	$\begin{pmatrix}0.79 & 0.19 & 0.01 & 0.00 & 0.00\\0.05 & 0.79 & 0.15 & 0.01 & 0.00\\0.00 & 0.16 & 0.71 & 0.11 & 0.01\\0.00 & 0.03 & 0.34 & 0.57 & 0.05\\0.01 & 0.04 & 0.21 & 0.45 & 0.26\end{pmatrix}$	$\begin{pmatrix}0.72 & 0.24 & 0.03 & 0.00 & 0.00\\0.06 & 0.73 & 0.19 & 0.02 & 0.00\\0.01 & 0.12 & 0.72 & 0.14 & 0.01\\0.00 & 0.02 & 0.18 & 0.71 & 0.09\\0.00 & 0.00 & 0.03 & 0.24 & 0.68\end{pmatrix}$	$\begin{pmatrix}0.74 & 0.24 & 0.01 & 0.00 & 0.00\\0.05 & 0.85 & 0.09 & 0.00 & 0.00\\0.01 & 0.56 & 0.39 & 0.03 & 0.00\\0.04 & 0.39 & 0.39 & 0.15 & 0.01\\0.11 & 0.48 & 0.26 & 0.11 & 0.04\end{pmatrix}$
HL2DM		Quake 3	
Predicted	Empirical	Predicted	Empirical
$\begin{pmatrix}0.68 & 0.27 & 0.04 & 0.00 & 0.00\\0.06 & 0.69 & 0.22 & 0.03 & 0.00\\0.01 & 0.12 & 0.70 & 0.16 & 0.01\\0.00 & 0.02 & 0.18 & 0.69 & 011\\0.00 & 0.00 & 003 & 0.24 & 0.68\end{pmatrix}$	$\begin{pmatrix}0.62 & 0.34 & 0.03 & 0.01 & 0.00\\0.07 & 0.75 & 0.15 & 0.02 & 0.00\\0.01 & 0.32 & 0.55 & 0.10 & 0.01\\0.01 & 0.16 & 0.47 & 0.31 & 0.05\\0.01 & 0.14 & 0.31 & 0.32 & 0.17\end{pmatrix}$	$\begin{pmatrix}0.54 & 0.35 & 0.09 & 0.01 & 0.01\\0.09 & 0.57 & 0.28 & 0.05 & 0.00\\0.02 & 0.18 & 0.57 & 0.21 & 0.03\\0.03 & 0.05 & 0.27 & 0.55 & 0.13\\0.00 & 0.01 & 0.09 & 0.34 & 0.50\end{pmatrix}$	$\begin{pmatrix}0.54 & 0.39 & 0.04 & 0.00 & 0.00\\0.01 & 0.66 & 0.27 & 0.05 & 0.00\\0.03 & 0.40 & 0.47 & 0.18 & 0.03\\0.00 & 0.30 & 0.47 & 0.17 & 0.02\\0.00 & 0.40 & 0.48 & 0.08 & 0.04\end{pmatrix}$
HLCS		HLDM	
Predicted	Empirical	Predicted	Empirical
$\begin{pmatrix}0.72 & 0.24 & 0.03 & 0.00 & 0.00\\0.06 & 0.73 & 0.19 & 0.02 & 0.00\\0.01 & 0.12 & 0.72 & 0.14 & 0.01\\0.00 & 0.02 & 0.18 & 0.71 & 0.09\\0.00 & 0.00 & 0.00 & 0.00 & 0.00\end{pmatrix}$	$\begin{pmatrix}0.74 & 0.24 & 0.01 & 0.00 & 0.00\\0.05 & 0.85 & 0.09 & 0.00 & 0.00\\0.01 & 0.56 & 0.39 & 0.03 & 0.00\\0.04 & 0.39 & 0.39 & 0.15 & 0.01\\0.11 & 0.48 & 0.26 & 0.11 & 0.04\end{pmatrix}$	$\begin{pmatrix}0.61 & 0.31 & 0.06 & 0.01 & 0.00\\0.08 & 0.63 & 0.25 & 0.04 & 0.00\\0.01 & 0.16 & 0.63 & 0.18 & 0.02\\0.00 & 0.03 & 0.23 & 0.62 & 0.12\\0.00 & 0.00 & 0.06 & 0.30 & 0.59\end{pmatrix}$	$\begin{matrix}0.44 & 0.49 & 0.04 & 0.01 & 0.01\\0.07 & 0.78 & 0.12 & 0.02 & 0.01\\0.02 & 0.48 & 0.41 & 0.08 & 0.01\\0.01 & 0.31 & 0.39 & 0.23 & 0.05\\0.04 & 0.47 & 0.24 & 0.14 & 0.09\end{matrix}$

$$\binom{N-n_1}{n_2-n_1+l}(1-p_{00})^{n_2-n_1+l}\,p_{00}{}^{N-n_2-l} \qquad (6)$$

The two events are independent so the probability of l and m sources making the transition is given by the product

$$\binom{n_1}{l}(1-p_{11})^l\,p_{11}{}^{n_1-l}\binom{N-n_1}{n_2-n_1+l} \\ .(1-p_{00})^{n_2-n_1+l}\,p_{00}{}^{N-n_2-l} \qquad (7)$$

The events described by the different values of l and m are mutually exclusive, so the elements of the transition matrix A giving the transition probability from n_1 sources generating packet lengths of length L to n_2 sources generating packet length of L is given by:

$$a(n_1,n_2) = \qquad (8)$$
$$\sum_{k=0}^{n_1}\left[\begin{matrix}\binom{n_1}{k}(1-p_{11})^k\,p_{11}{}^{n_1-k}. \\ \binom{N-n_1}{n_2-n_1+k}. \\ (1-p_{00})^{n_2-n_1+k}\,p_{00}{}^{N-n_2-k}\end{matrix}\right]$$

We can now compare the transition matrices generated by this expression with those derived empirically. Table 1 contains predicted and empirically obtained transition matrices for the 5 player games for ETPRO, HL2CS, HL2DM and Quake 3 respectively. Element i, j of the matrix shows the probability of the server generating a packet whose length falls into a bin size of jL given that they have just generated a packet whose length falls into a bin size of iL.

Despite the simplifying assumptions made in developing the models we see pleasing agreement between the predicted and empirically obtained transition probabilities. Agreement is particularly good for the smaller packet lengths and reasonably satisfactory for larger lengths for which there is less empirical data. Since packet length distributions for the 2-player games are negatively skewed, there is far more data describing transitions from the smaller bin sizes than transitions from the larger bin sizes with a consequent loss in accuracy as i increases.

In the next section we show how we can use this approach to simulate correlated packet length sequences for an N-player game when only the data for the 2 player game is known.

C. *Simulation Based on Synthetic Markov Chain Models*

We implemented a traffic generator, based on the model described in this paper, using the INET Framework [21] of the OMNeT++ [22] simulation environment. The traffic generator can model the packet inter-arrival times and the client to server packet size; however, this discussion will focus on the server to client packet size. For space reasons, the results from various simulations using the traffic generator are only presented for ETPRO and HL2DM games.

The server to client packet length cumulative distribution functions (CDFs) for ETPRO and HL2DM are shown in Figure 3. The results for both the 5 player ETPRO and 5 player HL2DM game show that the simulated packet length distribution is similar to the empirical distribution. In the case of the 7 and 9 player games, although still similar, the simulated results deviate more from the empirical results. The differences between the predicted and empirical state transition matrices are responsible for the differences between the simulated and empirical results. The expected relative frequencies of the packet sizes can be determined from the state transition matrix by calculating the long run probabilities.

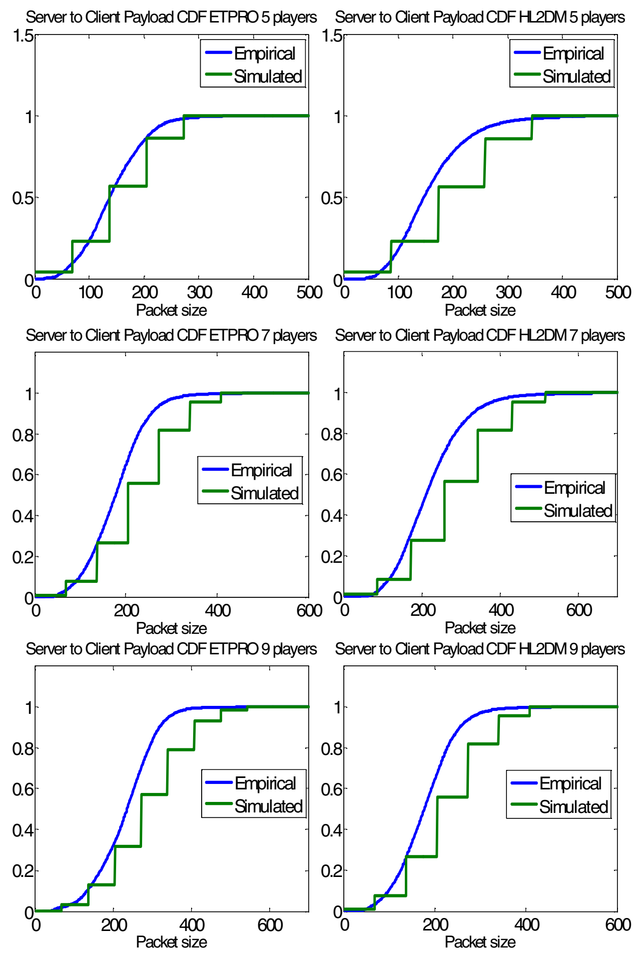

Figure 3. Empirical and simulated cumulative distribution function for ETPRO and HL2DM 5, 7 and 9-player games

The traffic generator produces "quantized" payload sizes that are multiples of median payload size for the two player game. The true payload sizes are not quantized. Nevertheless, this approach does approximate packet length reasonably well.

V. Conclusion and Future Research

In this paper we have shown that FPS game traffic is satisfactorily modeled by Markov processes, and we have shown how to extrapolate a Markov chain describing a simplified 2-player game to that of games with larger numbers of players. Finally we have shown how this modeling can be used to develop Markov Chain based simulations.

Future research will involve application of the techniques described to other game genres and application of the models to investigate game related network performance issues.

References

[1] M. Borella, "Source models of network game traffic," *Computer Communications,* vol. 23, pp. 403-410, Feb 2000.

[2] P. Branch and G. Armitage, "Extrapolating server to client IP traffic from empirical measurements of first person shooter games," in *5th Workshop on Network System Support for Games 2006 (Netgames2006)* Singapore, 2006.

[3] P. Branch and G. Armitage, "Measuring the auto-correlation of server to client traffic in first person shooter games," in *Australian Telecommunications, Network and Applications Conference (ATNAC)* Melbourne, Australia, 2006.

[4] Centre for Advanced Internet Architectures (CAIA), "Simulating online network games (SONG)," Accessed 14 March 2007, http://caia.swin.edu.au/sitcrc

[5] C. Chambers, W.-C. Feng, S. Sahu, and D. Saha, "Measurement-based characterization of a collection of on-line games," in *Internet Measurement Conference 2005 (IMC2005)* Berkeley California, 2005.

[6] J. Farber, "Traffic modelling for fast action network games," *Multimedia Tools and Applications,* vol. 23, pp. 31-46, Dec 22 2004.

[7] W.-C. Feng, F. Chang, W.-C. Feng, and J. Walpole, "A traffic charaterization of popular on-line games," *IEEE/ACM Transactions on Networking (TON),* vol. 13, pp. 488-500, June 2005.

[8] W.-C. Feng, F. Chang, W.-C. Feng, and J. Walpole, "Provisioning on-line games: A traffic analysis of a busy Counter-Strike server," in *SIGCOMM Internet Measurement Workshop,* Marseille, France, 2002.

[9] T. Henderson and S. Bhatti, "Modelling user behaviour in networked games," in *9th ACM International Conference on Multimedia (ACM Multimedia)* Ottawa, Canada, 2001.

[10] T. Henderson and S. Bhatti, "Networked games - a QoS-sensitive application for QoS insensitive users?," in *ACM SIGCOMM workshop on Revisiting IP QoS* Karlsruhe, Germany, 2003.

[11] T. Lang and G. Armitage, "A ns2 model for the Xbox system link game HALO," in *Australian Telecommunications, Networks and Applications Conference (ATNAC)* Melbourne, Australia, 2003.

[12] T. Lang, G. Armitage, P. Branch, and H.-Y. Choo, "A synthetic traffic model for Half-Life," in *Australian Telecommunications, Networks and Applications Conference (ATNAC)* Melbourne, 2003.

[13] T. Lang, P. Branch, and G. Armitage, "A synthetic model for Quake III traffic," in *Advances in Computer Entertainment (ACE2004)* Singapore, 2004.

[14] S. Zander and G. Armitage, "A traffic model for the XBOX game Halo 2," in *15th ACM International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV2005)* Washington, 2005.

[15] G. Armitage, M. Claypoole, and P. Branch, *Networking and online games: Understanding and engineering multiplayer Internet games*. Chichester, England: John Wiley and Sons Ltd, 2006.

[16] idsoftware, "id software downloads," Accessed 10 September 2007, http://www.idsoftware.com/business/techdownloads/

[17] G. Box and G. Jenkins, *Time series analysis: Forecasting and control*, revised ed. San Francisco: Holden-Day Inc, 1976.

[18] C. Chatfield, *The analysis of time series*. Bath, UK: Chapman and Hall/CRC, 2004.

[19] K. Elsayed and H. Perros, "A computationally efficient algorithm for characterizing the superposition of multiple heterogeneous interrupted Bernoulli processes," in *Second International Workshop on Numerical Solution of Large Markov Chains* Raleigh, North Carolina, 1995.

[20] J. Kemeney and J. L. Snell, *Finite Markov chains*. New York: Springer-Verlag, 1976.

[21] "INET Framework," Accessed February. 2007, http://www.omnetpp.org

[22] "OMNeT++ object-oriented discrete event simulation system," Accessed February 2007, http://www.omnetpp.org