Chapter 4

INTERNET CONGESTION: A LABORATORY EXPERIMENT

Daniel Friedman*
University of California, Santa Cruz

Bernardo Huberman
Hewlett-Packard Laboratories

Abstract

Human players and automated players (bots) interact in real time in a congested network. A player's revenue is proportional to the number of successful "downloads" and his cost is proportional to his total waiting time. Congestion arises because waiting time is an increasing random function of the number of uncompleted download attempts by all players. Surprisingly, some human players earn considerably higher profits than bots. Bots are better able to exploit periods of excess capacity, but they create endogenous trends in congestion that human players are better able to exploit. Nash equilibrium does a good job of predicting the impact of network capacity and noise amplitude. Overall efficiency is quite low, however, and players overdissipate potential rents, i.e., earn lower profits than in Nash equilibrium.

1. INTRODUCTION

The Internet suffers from bursts of congestion that disrupt cyberspace markets. Some episodes, such as gridlock at the Victoria's Secret site after a Superbowl advertisement, are easy to understand, but other episodes seem to come out of the blue. Of course, congestion is also important in many other contexts. For example, congestion sometimes greatly degrades the value of freeways, and in extreme cases (such as burning nightclubs) congestion can be fatal. Yet the dynamics of congestion are still poorly understood, especially when (as on the Internet) humans interact with automated agents in real time.

In this paper we study congestion dynamics in the laboratory using a multiplayer interactive video game called StarCatcher. Choices are real-time (i.e., asynchronous):

at every instant during a two minute period, each player can start to download or abort an uncompleted download. Human players can freely switch back and forth between manual play and a fully automated strategy. Other players, called bots, are always automated. Players earn revenue each time they complete the download, but they also accumulate costs proportional to waiting time.

Congestion arises because waiting time increases stochastically in the number of pending downloads. The waiting time algorithm is borrowed from Maurer and Huberman (2001), who simulate bot-only interactions. This study and earlier studies show that congestion bursts arise from the interaction of many bots, each of whom reacts to observed congestion observed with a short lag. The intuition is that bot reactions are highly correlated, leading to non-linear bursts of congestion.

At least two other strands of empirical literature relate to our work. Ochs (1990), Rapoport et al. (1998) and others find that human subjects are remarkably good at coordinating entry into periodic (synchronous) laboratory markets subject to congestion. More recently, Rapoport et al. (2003) and Seale et al. (2003) report fairly efficient queuing behavior in a laboratory game that has some broad similarities to ours, but (as discussed in section 5 below) differs in numerous details.

A separate strand of literature considers asynchronous environments, sometimes including bots. The Economist (2002) mentions research by Dave Cliff at HP Labs Bristol intended to develop bots that can make profits in major financial markets that allow asynchronous trading. The article also mentions the widespread belief that automated trading strategies provoked the October 1987 stock market crash. Eric Friedman et al. (forthcoming) adapt periodic laboratory software to create a near-asynchronous environment where some subjects can update choices every second; other subjects are allowed to update every 2 seconds or every 30 seconds. The subjects play quantity choice games (e.g., Cournot oligopoly) in a very low information environment: they know nothing about the structure of the payoff function or the existence of other players. Play tends to converge to the Stackelberg equilibrium (with the slow updaters as leaders) rather than to the Nash equilibrium. In our setting, by contrast, there is no clear distinction between Stackelberg and Nash, subjects have asynchronous binary choices at endogenously determined times, and they compete with bots.

After describing the laboratory set up in the next section, we sketch theoretical predictions derived mainly from Nash equilibrium. Section 4 presents the results of our experiment. Surprisingly, some human players earn considerably higher profits than bots. Bots are better able to exploit periods of excess capacity, but they create endogenous trends in congestion that human players are better able to exploit. The comparative statics of pure strategy Nash equilibrium do a good job of predicting the impact of network capacity and noise amplitude. However, overall efficiency is quite low relative to pure strategy Nash equilibrium, i.e., players "overdissipate" potential rents.

Section 5 offers some perspectives and suggestions for follow up work. Appendix A collects the details of algorithms and mathematical derivations. Appendix B reproduces the written instructions to human subjects.

2. THE EXPERIMENT

The experiment was conducted at UCSC's LEEPS lab. Each session lasts about 90 minutes and employs at least four human subjects, most of them UCSC undergraduates. Students sign up on line after hearing announcements in large classes, and are notified by email about the session time and place, using software developed by UCLA's CASSEL lab. Subjects read the instructions attached in Appendix B, view a projection of the user interface, participate in practice periods, and get public answers to their questions. Then they play 16 or more periods of the StarCatcher game. At the end of the session, subjects receive cash payment, typically \$15 to \$25. The payment is the total points earned in all periods times a posted payrate, plus a \$5.00 show-up allowance.

Each StarCatcher period lasts 240 seconds. At each instant, any idle player can initiate a service request by clicking the Download button, as in Figure 1. The service delay, or latency λ, is determined by an algorithm sketched in the paragraph after next. Unless the download is stopped earlier, after λ seconds the player's screen flashes a gold star and awards her 10 points. However, each second of delay costs the player 2 points, so she loses money on download requests with latencies greater than 5 seconds. The player can't begin a second download while an earlier request is still being processed but she can click the Stop button; to prevent excessive losses the computer automatically stops a request after 10 seconds. The player can also click the Reload button, which is equivalent to Stop together with an immediate new download request, and can toggle between manual mode (as just described) and automatic mode (described below).

The player's timing decision is aided by a real-time display showing the results of all service requests terminating in the previous 10 seconds. The player sees the mean latency as well as a latency histogram that includes Stop orders, as illustrated in Figure 1.

The delay algorithm is a noisy version of a single server queue model known in the literature as M/M/1. Basically, the latency λ is proportional to the reciprocal of current idle capacity. For example, if capacity is 6 and there are currently 4 active users, then the delay is proportional to $1/(6 - 4) = {}^1\!/_2$. In this example, 5 users would double the delay and 6 users would make the delay arbitrarily long. As explained in Appendix A, the actual latency experienced by a user is modified by a mean reverting noise factor, and is kept positive and finite by truncating at specific lower and upper bounds.

The experiments include automated players (called robots or bots) as well as humans. The basic algorithm for such players is: initiate a download whenever the mean latency (shown on all players' screens) is less than 5 seconds minus a tolerance, i.e., whenever it seems sufficiently profitable. The tolerance averages 0.5 seconds, corresponding to an intended minimum profit margin of 1 point per download. Appendix A presents details of the algorithm. Human players in most sessions have the option of "going on autopilot" using this algorithm, as indicated by the toggle button in Figure 1 Go To Automatic / Go To Manual. Subjects are told,

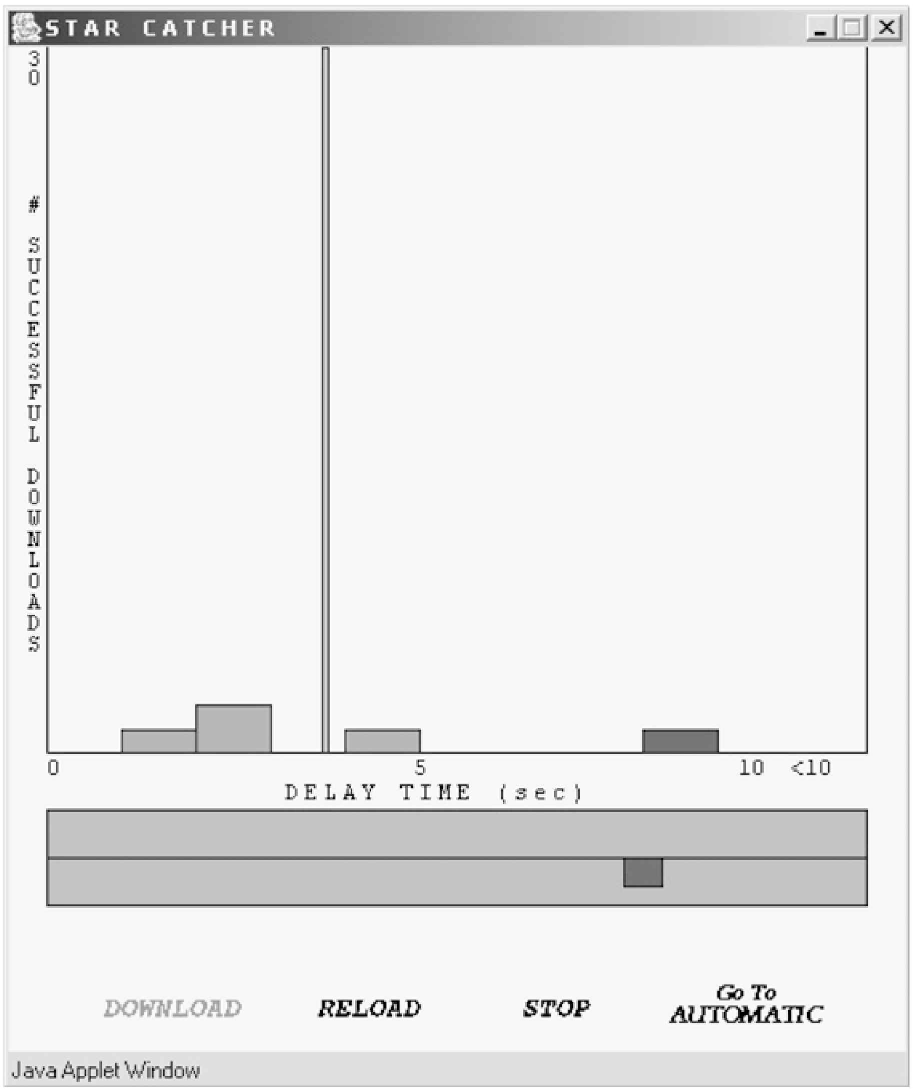


Figure 1. *User interface. The four decision buttons appear at the bottom of the screen; the download button is faded because the player is currently waiting for his download request to finish. The dark box on the thick horizontal bar just above the decision buttons indicates an 8 second waiting time (hence a net loss) so far. The histogram above reports results of download requests from all players terminating in the last 10 seconds. Here, one download took 2 seconds, two took 3 seconds, one took 5 seconds and one took 9 seconds. The color of the histogram bar indicates whether the net payoff from the download was positive (green, here light grey) or negative (red, here dark grey). The thin vertical line indicates the mean delay, here about 3.7 seconds. Time remaining is shown in a separate window.*

When you click GO TO AUTOMATIC a computer algorithm decides for you when to download. There sometimes are computer players (in addition to your fellow humans) who are always in AUTOMATIC. The algorithm mainly looks at the level of recent congestion and downloads when it is not too large.

The network capacity and the persistence and amplitude of the background noise is controlled at different levels in different periods. The number of human players and bots also varies; the humans who are sidelined from StarCatcher for a few periods use the time to play an individual choice game such as TreasureHunt, described in Friedman et al. (2003). Table 1 summarizes the values of the control variables used in all sessions analyzed below.

3. THEORETICAL PREDICTIONS

A player's objective each period is to maximize profit $\Pi = rN - cL$, where r is the reward per successful download, N is the number of successful downloads, c is the delay cost per second, and L is the total latency time summed over all download attempts in that period. The relevant constraints include the total time T in the period, and the network capacity C. The constant of proportionality for latency, i.e., the time scale S, is never varied in our experiments.

An important benchmark is social value V^*, the maximized sum of players' profits. That is, V^* is the maximum total profit obtainable by an omniscient planner who controls players' actions. Appendix A shows that, ignoring random noise, that benchmark is given by the expression $V^* = 0.25S^{-1}Tr(1 + C - cS/r)^2$. Typical parameter values in the experiment are $T = 120$ seconds, $C = 6$ users, $S = 8$ user-sec, $c = 2$ points/sec and $r = 10$ points. The corresponding social optimum values are $U^* = 2.70$ active users, $\lambda^* = 1.86$ seconds average latency, $\pi^* = 6.28$ points per download, $N^* = 174.2$ downloads, and $V^* = 1094$ points per period.

Of course, a typical player tries to increase his own profit, not social value. A selfish and myopic player will attempt to download whenever the incremental apparent profit π is sufficiently positive, i.e., whenever the reward $r = 10$ points sufficiently exceeds the cost λc at the currently displayed average latency λ. Thus such a player will choose a latency threshold ε and follow

Rule R. If idle, initiate a download whenever $\lambda \le r/c - \varepsilon$.

In Nash equilibrium (NE) the result typically will be inefficient congestion, because an individual player will not recognize the social cost (longer latency times for everyone else) when choosing to initiate a download. Our game has many pure strategy NE due to the numerous player permutations that yield the same overall outcome, and due to integer constraints on the number of downloads. Fortunately, the NE are clustered and produce outcomes in a limited range.

To compute the range of total NE total profit V^{NE} for our experiment, assume that all players use the threshold $\varepsilon = 0$ and assume again that noise is negligible. No

Table 1. Design of Sessions

Date	*# of periods*	*# of player-periods*											*Max # of robots*	*Max # human players*	*Experienced humans*
		Total	*By volatility*		*By capacity*										
			low	*high*	*2*	*3*	*4*	*5*	*6*	*7*	*9*				
8/21/02	27	159	87	72	20	101	38	0	0	0	0		4	4	no
8/22/02	32	189	94	95	19	120	50	0	0	0	0		4	4	no
8/20/02	32	192	97	95	21	117	54	0	0	0	0		4	4	yes
9/11/02	32	243	130	113	0	56	90	0	97	0	0		0	6	yes
9/12/02	32	199	101	98	20	126	53	0	0	0	0		5	3	yes
9/5/02	32	193	99	94	20	121	52	0	0	0	0		4	4	no
1/24/03	16	127	54	73	0	37	40	0	50	0	0		4	6	no
1/31/03	24	155	77	78	20	105	30	0	0	0	0		4	4	no
2/5/03	27	216	120	96	0	54	72	0	90	0	0		4	6	yes
2/4/03	16	104	52	52	10	64	30	0	0	0	0		4	4	no
2/12/03	18	143	71	72	0	36	47	0	60	0	0		4	6	no
2/14/03	27	214	119	95	0	54	72	0	88	0	0		4	6	no
2/19/03	31	194	100	94	19	123	52	0	0	0	0		4	4	yes
5/23/03	27	164	89	75	0	94	64	6	0	0	0		5	6	no
10/2/03	31	112	63	49	76	22	6	8	0	0	0		4	4	no
10/3/03	27	164	86	78	17	45	50	8	0	20	24		6	6	no

Volatility: low: Sigma = .0015, Tau = .0002; Volatility: high: Sigma = .0025, Tau = .00002 player will earn negative profits in NE, since the option is always available to remain idle and earn zero profit. Hence the lower bound on V^{NE} is zero. Appendix A derives the upper bound $V^{MNE} = T(rC - cS)/S$ from the observation that it should never be possible for another player to enter and earn positive profits. Hence the maximum NE efficiency is $V^{MNE}/V^* = 4(C - cS/r)/\,(1 + C - cS/r)^2 = 4U^{MNE}/(1 + U^{MNE})^2$. For the parameter values used above ($T = 120$, $C = 6$, $S = 8$, $c = 2$ and $r = 10$), the upper bound NE values are $U^{MNE} = 4.4$ active users (players), $\lambda^{MNE} = 3.08$ seconds delay, $\pi^{MNE} = 3.85$ points per download, $N^{MNE} = 171.6$ downloads, and $V^{MNE} = 660.1$ points per period, for a maximum efficiency of 60.4%.

The preceding calculations assume that the number of players m in the game is at least $U^{MNE} + 1$, so that congestion can drive profit to zero. If there are fewer players, then in Nash equilibrium everyone is always downloading. In this case there is excess capacity $a = U^{MNE} + 1 - m = C + 1 - cS/r - m > 0$ and, as shown in the Appendix, the interval of NE total profit shrinks to a single point, $\Pi m = Tram/S$.

What happens if the background noise is not negligible? As explained in the Appendix, the noise is mean-reverting in continuous time. Thus there will be some good times when effective capacity is above C and some bad times when it is lower. Since the functions V^{MNE} and V^* are convex in C (and bounded below by zero), Jensen's inequality tells us that the loss of profit in bad times does not fully offset the gain in good times. When C and m are sufficiently large (namely, $m > C > cS/r + 1$, where the last expression is 2.6 for the parameters above), this effect is stronger for V^* than for V^{MNE}. In this case Nash equilibrium efficiency V^{MNE}/V^* decreases when there is more noise. Thus the prediction is that aggregate profit should increase but that efficiency should decrease in the noise amplitude $\sigma/\sqrt{2\tau}$ (see Appendix A).[1]

A key testable prediction arises directly from the Nash equilibrium benchmarks. The null hypothesis, call it full rent dissipation, is that players' total profits will be in the Nash equilibrium range. That is, when noise amplitude is small, aggregate profits will be $V^{MNE} = Tram/S$ in periods with excess capacity $a > 0$, and will be between 0 and $V^{MNE} = T(rC - cS)/S$ in periods with no excess capacity. The corresponding expressions for efficiency have already been noted.

One can find theoretical support for alternative hypotheses on both sides of the null. Underdissipation refers to aggregate profits higher than in any Nash equilibrium, i.e., above V^{MNE}. This would arise if players can maintain positive thresholds ε in Rule R, for example. A libertarian justification for the underdissipation hypothesis is that players somehow self-organize to partially internalize the congestion externality (see e.g., Gardner, Ostrom, and Walker, 1992). For example, players may discipline each other using punishment strategies. Presumably the higher profits would emerge in later periods as self-organization matures. An alternative justification from behavioral economics is that players have positive regard for the other players' utility of payoffs, and will restrain themselves from going after the last penny of personal profits in order to reduce congestion. One might expect this effect to weaken a bit in later periods.

Overdissipation of rent, i.e., negative aggregate profits, is the other possibility. One theoretical justification is that players respond to relative payoff and see increasing returns to downloading activity (e.g., Hehenkamp et al., 2001). A behavioral economics justification is that people become angry at the greed of other players and are willing to pay the personal cost of punishing them by deliberately increasing congestion (e.g., Cox and Friedman, 2002). Behavioral noise is a third possible justification. For example, Anderson, Goeree and Holt (1998) use quantal response equilibrium, in essence Nash equilibrium with behavioral noise, to explain overdissipation in all-pay auctions.

Further insights may be gained from examining individual decisions. The natural null hypothesis is that human players follow Rule R with idiosyncratic values of the threshold ε. According to this hypothesis, the only significant explanatory variable for the download decision will be $\lambda - r/c = \lambda - 5$ sec, where λ is the average latency currently displayed on the screen. An alternative hypothesis (which occurred to us only after looking at the data) is that some humans best-respond to Rule R behavior, by anticipating when such behavior will increase or decrease λ and reacting to the anticipation.

The experiment originally was motivated by questions concerning the efficiency impact of automated Rule R strategies. The presumption is that bots (and human players in auto mode) will earn higher profits than humans in manual mode.[2] How strong is this effect? On the other hand, does a greater prevalence of bots depress everyone's profit? If so, is the second effect stronger than the first, i.e., are individual profits lower when everyone is in auto mode than when everyone is in manual mode? The simulations reported in Maurer and Huberman (2001) confirm the second effect but disconfirm the social dilemma embodied in the last question. Our experiment examines whether human subjects produce similar results.

4. RESULTS

We begin with a qualitative overview of the data. Figure 2 below shows behavior in a fairly typical period. It is not hard to confirm that bots indeed follow the variable λ = average delay: their download requests cease when λ rises above 4 or 5, and the line indicating the number of bots downloading stops rising. It begins to decline as existing downloads are completed. Likewise, when λ falls below 4 or 5, the number of bot downloads starts to rise.

The striking feature about Figure 2 is that the humans are different. They appear to respond as much to the *change in* average delay. Sharp decreases in average delay encourage humans to download. Perhaps they anticipate further decreases, which would indeed be likely if most players use Rule R. We shall soon check this conjecture more systematically.

Figure 3 shows another surprise, strong overdissipation. Both bots and humans lose money overall, especially bots (which include humans in the auto mode). The top half of human players spend only 1% of their time in auto mode, and even the bottom half spend only 5% of their time in auto mode. In manual mode, bottom half human players lose lots of money but at only 1/3 the rate of bots, and top half humans actually make modestly positive profit.

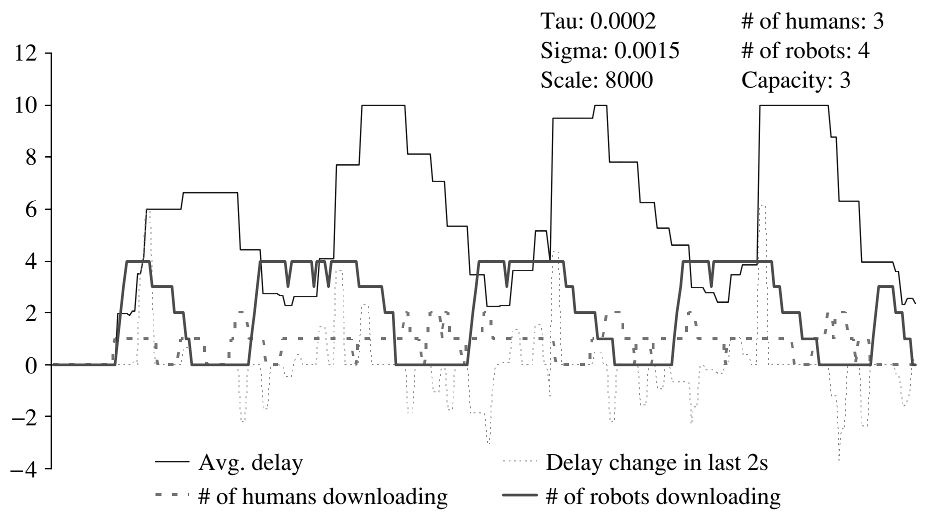


Figure 2. Exp. 09-12-2002, Period 1.

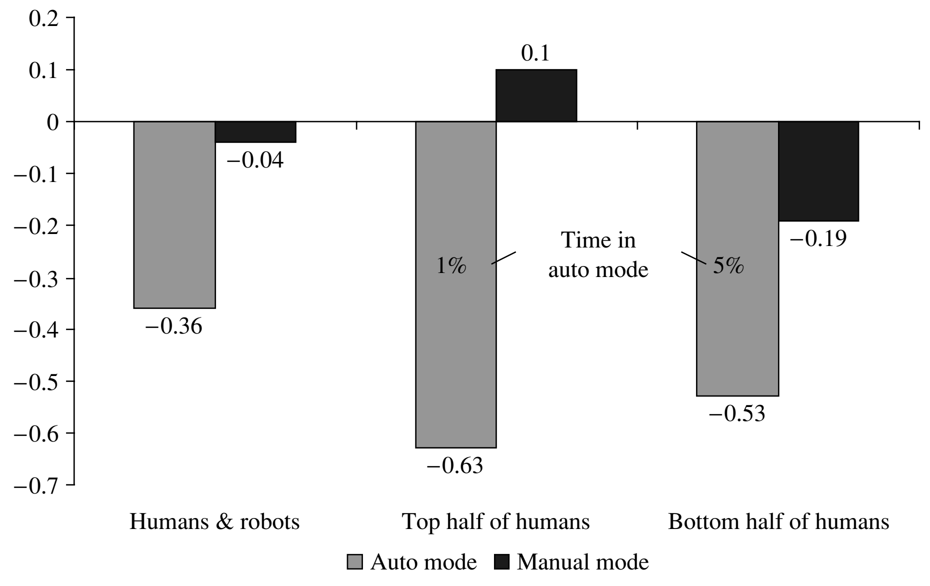


Figure 3. Profit per second in auto and manual mode.

Figure 4 offers a more detailed breakdown. When capacity is small, there is only a small gap between social optimum and the upper bound aggregate profit consistent with Nash Equilibrium, so Nash efficiency is high as shown in the green bars for $C = 2, 3, 4$. Bots lose money rapidly in this setting because congestion sets in

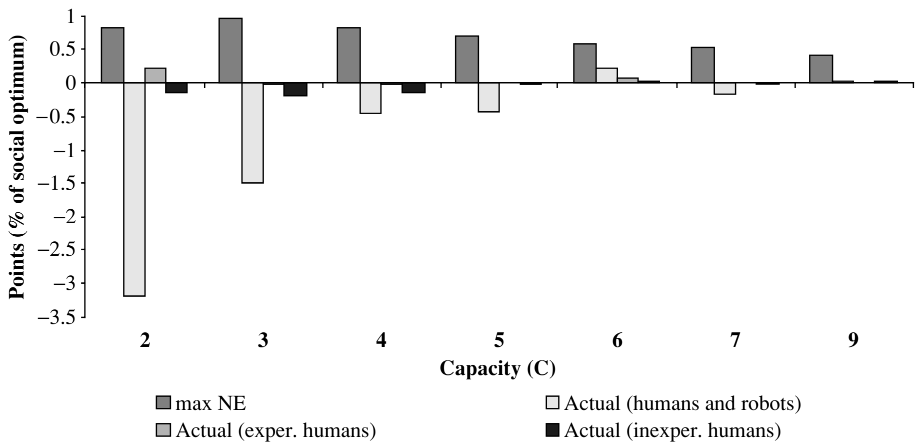


Figure 4. Theoretical and actual profits as percentage of social optimum.

quickly when capacity is small. Humans lose money when inexperienced. Experienced human players seem to avoid auto mode and learn to anticipate the congestion sufficiently to make positive profits. When capacity is higher ($C = 6$), bots do better even than experienced humans, perhaps because they are better at exploiting the good times with excess capacity. (Of course, overdissipation is not feasible with excess capacity: in NE everyone downloads as often as physically possible and everyone earns positive profit.)

We now turn to more systematic tests of hypotheses. Table 2 below reports OLS regression results for profit rates (net payoff per second) earned by four types of players. The first column shows that bots (lumped together with human players in auto mode) do much better with larger capacity and with higher noise amplitude, consistent with NE predictions. The effects are highly significant, statistically as well as economically. The other columns indicate that humans in manual mode are able to exploit increases in capacity only about half as much as bots, although the effect is still statistically highly significant for all humans and top half of humans. The next row suggests that bots but not humans are able to exploit higher amplitude noise. The last row of coefficient estimates finds that, in our mixed bot-human experiments, the interaction [noise amplitude with excess fraction of players in auto mode] has the opposite effect for bots as in Maurer and Huberman (2001), and has no significant effect for humans.

Table 3 above reports a fine-grained analysis of download decisions, the dependent variable in the logit regressions. Consistent with Rule R (hardwired into their algorithm), the bots respond strongly and negatively to the average delay observed on the screen minus $r/c = 5$. Surprisingly, the regression also indicates that bots are more likely to download when the observed delay increased over the last 2 seconds; we interpret this as an artifact of the cyclical congestion patterns. Fortunately

Table 2. OLS Estimates of profit rates

Indep. variables	*For:*	*Auto mode All players*	*All humans*	*Manual mode Top half*	*Bottom half*
Intercept		0.88	0.48	0.64	not sig.
Excess capacity		0.69	0.27	0.29	0.17[a]
Excess capacity^2		0.08	0.03	0.04	not sig.
Noise		0.53	not sig.	not sig.	−0.12[b]
Noise*(s – 1/2)		−1.81	not sig.	not sig.	not sig.
NOBS		1676	1222	640	582

Notes: all significant at $p < 0.01$, except a: $p = 0.04$, b: $p = 0.06$
Excess capacity = $a = C - m - 0.6$
Noise = sigma/$\sqrt{(2\text{Tau})}$
s = fraction of all players in auto mode per period

Table 3. Logit regression for download decision

Indep. Variables	*For:*	*Auto mode All players*		*All humans*	*Manual mode Top half*	*Bottom half*
Intercept		−2.08	−2.12	−2.51	−2.31	−2.74
Avg. delay −5s		−0.31	−0.31	0.03	0.06	−0.01[a]
2s change in avg. delay		0.08		−0.19	−0.30	−0.08
NOBS		163,602	163,602	186,075	92,663	93,412

Notes: all significant at $p < 0.01$, except a: $p = 0.047$

the delay coefficient estimate is unaffected by omitting the variable for change in delay.

Human players react in the opposite direction to delay changes. The regressions confirm the impression gleaned from Figure 2 that humans are much more inclined to initiate download requests when the observed delay is decreasing. Perhaps surprisingly, experienced humans are somewhat more inclined to download when the observed delay is large. A possible explanation is that they then anticipate less congestion from bots.

The results reported above seem fairly robust to changes in the specification. In particular, including time trends within or across periods seems to have little systematic impact.

5. DISCUSSION

The most surprising result is that human players outperform the current generation of automated players (bots). The bots do quite badly when capacity is low. Their decision rule fails to anticipate the impact of other bots and neglects the difference between observed congestion (for recently completed download attempts) and anticipated congestion (for the current download attempt). Human players are slower and less able to exploit excess capacity (including transient episodes due to random noise), but some humans are far better at anticipating and exploiting the congestion trends that the bots create. In our experiment the second effect outweighs the first, so humans earn higher profits overall than bots.

Perhaps the most important questions in our investigation concerned rent dissipation. Would human players find some way to reduce congestion costs and move towards the social optimum, or would they perhaps create even more congestion than in Nash equilibrium? Sadly, overdissipation outcomes are most prevalent in our data.

The Nash comparative statics, on the other hand, generally help explain the laboratory data. Nash equilibrium profit increases in capacity and noise amplitude, and so do observed profits.

Several directions for future research suggest themselves. First, one might want to look at smarter bots. Preliminary results show that it is not as easy as we thought to find more profitable algorithms; linear extrapolation from available data seems rather ineffective. That project contemplates higher levels of sophistication (in a sense similar to Stahl and Wilson, 1995) but the results are not yet in.

Second, one might want to connect our research to the experiments on queuing behavior. As noted in the introduction, Rapoport et al. (2003) and a companion paper reported fairly efficient outcomes, rather different than our own. Which design differences from ours are crucial? The list of possible suspects is quite long: no bots; synchronous decisions in discrete time; a single service request per player each period; simultaneous choice at the beginning of the period; precommited requests (no counterpart to our “stop” or “reload”); deterministic and constant service times in a first-in, first-out queue; no information feedback during the period; and no information feedback between periods regarding congestion at times not chosen. Answering the question may not be easy, but it surely would be interesting.

More generally, one might want to probe the robustness of the overdissipation result. It clearly should be checked in humans-only and in bots-only environments, and preliminary results seem consistent with the findings reported above. One should also check alternative congestion functions to the mean-reverting noisy M/M/1 queuing process. Finally, it would be quite interesting to investigate mechanisms such as congestion taxes to see whether they enable humans and robots to earn healthier profits in congestible real-time environments.

NOTES

* The National Science Foundation funded this work under grant IIS-9986651. We are also grateful for the research assistance of Alessandra Cassar, Garrett Milam, Ralf Hepp, and Kai Pommerenke, and the programming support of Neil Macneale, Jeremy Avnet and Nitai Farmer. We benefited from comments by colleagues including Joshua Aizenman, Eileen Brooks, Rajan Lukose, Nirvikar Singh, and Donald Wittman, and by HKUST conference participants, especially Yan Chen, Rachel Croson and Eric Johnson. Significant improvements to the final version are due to the comments of an anonymous referee.

[1] The simulations reported in Maurer and Huberman (2001) suggest an alternative hypothesis: profits increase in noise amplitude times ($s - \frac{1}{2}$), where s is the fraction of players in auto mode. It should be noted that their bot algorithm supplemented Rule R with a Reload option.

[2] Indeed, a referee of our grant proposal argued that it was redundant to use human subjects. He thought it obvious that the bots would perform better.

[3] This variable is generated by summing up the times for successful (the download took less than or exactly ten seconds) and unsuccessful (failed download attempt, i.e., no download within ten seconds) download attempts that were *completed within the last ten seconds*. The result is then divided by the number of download attempts to lead to the average delay (*AD*). The variable is continuously updated. Times for download attempts that have been aborted (by the player hitting the "STOP" or the "RELOAD" button) are disregarded.

REFERENCES

Anderson, S., Goeree, J. and Holt, C., (August 1998). "The All-Pay Auction: Equilibrium with Bounded Rationality." *Journal of Political Economy*, 106(4), 828–853.

Cox J. C. and Friedman, D. (October 2002). "A Tractable Model of Reciprocity and Fairness," UCSC Manuscript.

Friedman, Eric, Mikhael Shor, Scott Schenker, and Barry Sopher, (November 30, 2002). "An Experiment on Learning with Limited Information: Nonconvergence, Experimentation Cascades, and the Advantage of Being Slow." *Games and Economic Behavior* (forthcoming) *Economist* magazine, "Robo-traders," p. 65.

Hehenkamp, Burkhard, Leininger, Wolfgang, and Possajennikov, Alex, (December 2001). "Evolutionary Rent Seeking." CESifo Working Paper 620.

Gardner, Roy, Ostrom, Elinor and Walker, James, (June 1992). "Covenants With and Without a Sword: Self-Governance is Possible." *American Political Science Review*, 86(2), 404–417.

Maurer, Sebastian and Bernardo Huberman, (2001). "Restart Strategies and Internet Congestion." *Journal of Economic Dynamics & Control* 25, 641–654.

Ochs, Jack, (May, 1990). "The Coordination Problem in Decentralized Markets: An Experiment." *The Quarterly Journal of Economics*, 105(2), 545–559.

Rapoport, A., Seale, D. A., Erev, I., & Sundali, J. A., (1998). "Equilibrium Play in Large Market Entry Games." *Management Science*, 44, 119–141.

Rapoport, A., Stein, W., Parco, J. and Seale, D., (July 2003). "Equilibrium Play in Single Server Queues with Endogenously Determined Arrival Times." University of Arizona Manuscript.

Seale, D., Parco, J., Stein, W. and Rapoport, A., (January 2003). "Joining a Queue or Staying Out: Effects of Information Structure and Service Time on Large Group Coordination." University of Arizona Manuscript.

Stahl, D. O. and Wilson, P., (1995). "On Players' Models of Other Players – Theory and Experimental Evidence." *Games and Economic Behavior*, 10, 213–254.

APPENDIX A. TECHNICAL DETAILS.

A.1. Latency and Noise. Following the noisy M/M/1 queuing model of Maurer and Huberman (2001), latency for a download request initiated at time t is

$$\lambda(t) = \frac{S[1 + e(t)]_+}{1 + C - U(t)} \tag{A1}$$

if the denominator is positive, and otherwise is $\lambda^{\max} > 0$. To unpack the expression (A1), note that the subscripted "+" refers to the positive part, i.e., $[x]_+ = \max\{x, 0\}$. The parameter C is the capacity chosen for that period; more precisely, to remain consistent with conventions in the literature, C represents full capacity minus 1. The parameter S is the time scale, or constant of proportionality, and $U(t)$ is usage, the number of downloads initiated but not yet completed at time t. The experiment truncates the latency computed from (A1) to the interval [0.2, 10.0] seconds. The lower truncation earns the 10 point reward but the upper truncation at $\lambda^{\max} = 10$ seconds does not.

The random noise $e(t)$ is Normally distributed with volatility σ and unconditional mean 0. The noise is mean reverting in continuous time and follows the Ornstein-Uhlenbeck process with persistence parameter $\tau > 0$ (see Feller, p. 336). That is, $e(0) = 0$ and, given the previous value $x = e(t - h)$ drawn at time $t - h > 0$, the algorithm draws a unit Normal random variate z and sets $e(t) = x\exp(-\tau h) + z\sigma\sqrt{[1 - \exp(-2\tau h)]/(2\tau)}$. Thus the conditional mean of noise is typically different from zero; it is the most recently observed value x shrunk towards zero via an exponential term that depends on the time lag h since the observation was made and a shrink rate $\tau > 0$. In the no-persistence (i.e., no mean reversion or shrinking) limit $\tau \to 0$, we have Brownian motion with conditional variance $\sigma^2 h$, and $e(t) = x + z\sigma\sqrt{h}$. In the long run limit as $h \to \infty$ we recover the unconditional variance $\sigma^2/(2\tau)$. The appropriate measure of noise amplitude in our setting therefore is its square root $\sigma/\sqrt{2\tau}$.

In our experiments we used two levels each for σ and τ. Rescaling time in seconds instead of milliseconds, the levels are 2.5 and 1.5 for σ, and 0.2 and 0.02 for τ. Figure A1 shows typical realizations of the noise factor $[1 + e(t)]_+$ for the two combinations used most frequently, low amplitude (low σ, high τ) and high amplitude (high σ, low τ).

A.2. Efficiency, no noise case. Social value V is the average net benefit $\pi = r - \lambda c$ per download times the total number of downloads $n \approx UT/\lambda$, where λ is the average latency, T is the length of a period and U is the average number of users attempting to download. Assume that $\sigma = 0$ (noise amplitude is zero) so by (A1) the average latency is $\lambda = S/(1 + C - U)$. Assume also that the expression for n is exact. Then the first order condition (taking the derivative of $V = \pi n$ with respect to U and finding the root) yields $U^* = 0.5(1 + C - cS/r)$. Thus $\lambda^* = 2S/(1 + C + cS/r)$, and so maximized social value is $V^* = 0.25S^{-1}Tr(1 + C - cS/r)^2$.

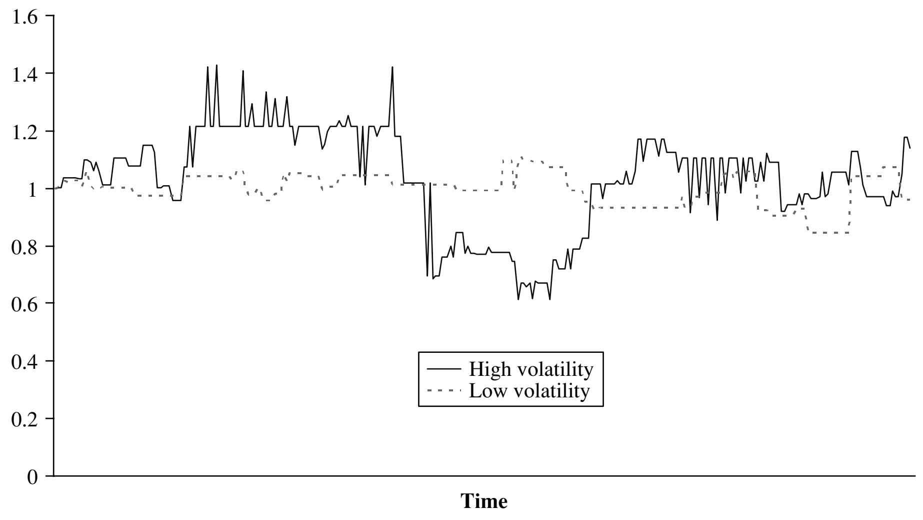


Figure A1. Noise (exp 10/3/03, periods 1 and 4).

To obtain the upper bound on social value consistent with Nash equilibrium, suppose that more than 10 seconds remain, the player currently is idle and the expected latency for the current download is λ. The zero-profit latency is derived from $0 = \pi = r - \lambda c$. Now $\lambda = r/c$ and the associated number of users is $U^{**} = 2U^* = C + 1 - cS/r$. Hence the minimum number of users consistent with NE is $U^{MNE} = U^{**} - 1 = C - cS/r$. The associated latency is $\lambda^{MNE} = rS/(r + cS)$, and the associated profit per download is $\pi^{MNE} = r^2/(r + cS)$, independent of C. The maximum number of downloads is $N^{MNE} = TU^{MNE}/\lambda^{MNE} = T(r + cS)(rC - cS)/(r^2S)$. Hence the upper bound on NE total profit is $V^{MNE} = N^{MNE}\pi^{MNE} = T(rC - cS)/S$, and the maximum NE efficiency is $V^{MNE}/V^* = (C - cS/r)/(1 + C - cS/r)^2 = 4U^{MNE}/(1 + U^{MNE})^2 \equiv Y$. Since $dU^{MNE}/dC = 1$, it follows that $dY/dC < 0$ iff $dY/dU^{MNE} < 0$ iff $1 < U^{MNE} = C - cS/r$. It is easy to verify that Y is $0(1/C)$.

A.3 Bot algorithm. In brief, the bot algorithm uses Rule R with a random threshold ε drawn independently from the uniform distribution on [0, 1.0] sec. The value of λ is the mean reported in the histogram window, i.e., the average for download requests completed in the last 10 seconds. Between download attempts the algorithm waits a random time drawn independently from the uniform distribution on [.25, .75] sec.

In detail, bots base their decision on whether to initiate a download on two factors. One of these determinants is the variable "average delay"[3] (*AD*). The second factor is a configurable randomly drawn threshold value. In each period, bots (and real players in automatic mode) have three behavior settings that can be set by the experimenter. If they aren't defined for a given period, then the previous settings are used, and if they are never set, then the default settings are used. An example (using the default settings) is

AutoBehavior Player 1: MinThreshold 4000, RandomWidth 1000, PredictTrend Disabled

The definitions are:

1) MinThreshold (*MT*): The lowest possible threshold value in milliseconds. If the average delay is below this minimum threshold, then there is 100% certainty that the robot (or player in Auto mode) will attempt a download if not already downloading. The default setting is 4000 (= 4 seconds).
2) Random Width (*RW*): The random draw interval width in milliseconds. This is the maximum random value that can be added to the minimum threshold value to determine the actual threshold value instance. That is, $MT + RW = Max\ Threshold\ Value$.
3) Predict Trend (*PT*): The default setting is Disabled. However, when Enabled, the following linear trend prediction algorithm is used: $MT_2 = MT + AD_2 - AD$. A new Minimum Threshold (MT_2) is calculated and used instead of the original Minimum Threshold value (*MT*). The average delay (*AD*) from exactly 2 seconds ago (AD_2) is used to determine the new Minimum Threshold value.

A bot will attempt a download when $AD \leq T = MT + RD$. A new threshold value (*T*) will be drawn (*RD* from a uniform distribution on [0, *RW*]) after each download attempt by the robot. Another important feature of the robot behavior is that a robot will never abort a download attempt.

To avoid artificial synchronization of robot download attempts, the robots check on *AD* every *x* seconds, where *x* is a uniformly distributed random variable on [.05, .15] seconds. Also, there is a delay (randomly picked from the uniform distribution on [.15, .45] seconds) after a download (successful or unsuccessful) has been completed and before the robot is permitted to download again. Both delays are drawn independently from each other and for each robot after each download attempt. The absolute maximum time a robot could wait after a download attempt ends and before initiating a new download (given that *AD* is sufficiently low) is thus 450ms + 150ms = 600ms.

APPENDIX B: STARCATCHER INSTRUCTIONS

UCSC 2/2003

I. GENERAL

You are about to participate in an experiment in the economics of interdependent decision-making. The National Science Foundation and other foundations have provided the funding for this project. If you follow these instructions carefully and make good decisions, you can earn a CONSIDERABLE AMOUNT OF MONEY, which will be PAID TO YOU IN CASH at the end of the experiment.

Your computer screen will display useful information regarding your payoffs and recent network congestion. Remember that the information on your computer screen is PRIVATE. In order to insure best results for yourself and accurate data for the experimenters, please do not communicate with the other participants at any point during the experiment. If you have any questions, or need assistance of any kind, raise your hand and somebody will come to you.

In the experiment you will interact with a group of other participants over a number of periods. Each period will last several minutes. In each period you earn “points” which are converted into cash at a pre-announced rate that is written on the board. You earn points by downloading stars. Each star successfully downloaded gives you 10 points, but waiting for a star to download incurs a cost. Every second that it takes to download the star will cost you 2 points. For example, if you start a download and it completes in 2 seconds, your delay cost is 4 = 2 points per second times 2 seconds. Therefore in this example you would earn 10 − 4 = 6 points.

Download delays range up to 10 seconds, depending on the number of other participants trying to download at the same time and background congestion. The delay cost can exceed the value of the download, so you can lose money when the network is congested. If the download takes 9 seconds you would earn 10 – 2*9 = −8 points, a negative payoff since the delay cost (18) is larger than the value of a star (10). Of course you can wait till the congestion clears: that way you don’t make money, but neither will you lose any. Doing nothing earns you zero, but also costs zero.

II. ACTIONS

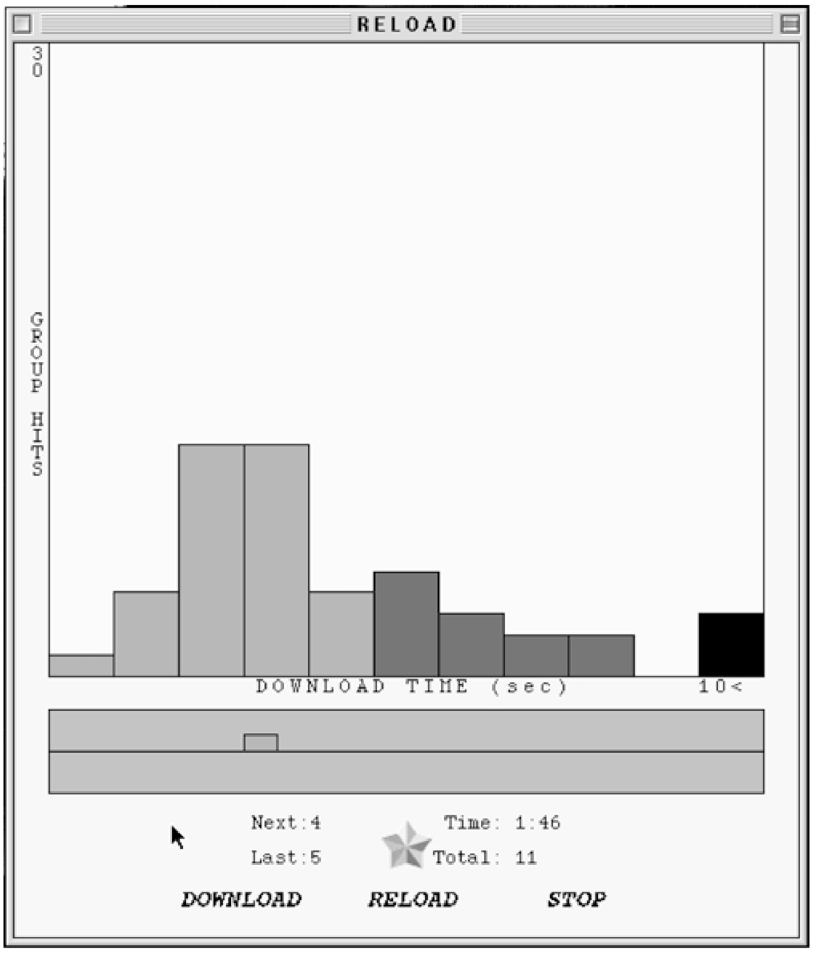
You have four action buttons: DOWNLOAD, RELOAD, STOP or GO TO AUTOMATIC. Clicking the DOWNLOAD button starts to download a star, and also starts to accumulate delay costs, until either:

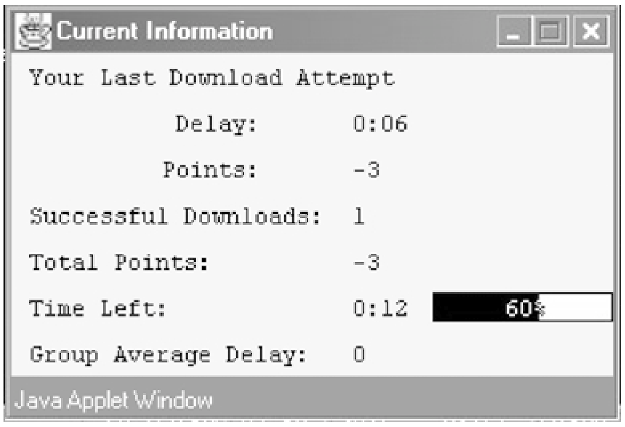
– The star appears on your screen, so you earn 10 points minus the delay cost; or
– The star does not appear within 10 seconds, so you lose 20 points; or
– You click the STOP button before 10 seconds elapse, so you lose twice the number of seconds elapsed; or
– You click the RELOAD button. This is like hitting STOP and DOWNLOAD immediately after.

When you click GO TO AUTOMATIC a computer algorithm decides for you when to download. There sometimes are computer players (in addition to your fellow humans) who are always in AUTOMATIC. The algorithm mainly looks at the level of recent congestion and downloads when it is not too large.

III. SCREEN INFORMATION

Your screen gives you useful information to help you choose your action. The main window reports congestion on the network (how many people were downloading) in the last 10 seconds. The horizontal axis shows the delay time (from 0 to 10 seconds) and the height of each vertical bar represent the number of successful downloads. For example, in the 10 seconds slice of history shown in Figure 1, one successful hit took one second, 4 successful hits took two seconds, 10 took three seconds, 10 took four seconds, 4 took five seconds, etc. The color of the bar indicates whether the payoff from the download was positive (green) or negative (red). The Black bar on the right indicates the number of people who waited unsuccessfully for a star. The Blue bar (not shown in picture) indicates the number of people who hit Stop or Reload.




Current Information

```
Your Last Download Attempt

            Delay:        0:06

           Points:        -3

Successful Downloads:     1

Total Points:             -3

Time Left:                0:12   60%

Group Average Delay:      0
```

Java Applet Window

Just below the graph showing recent traffic is a horizontal status bar. This "status bar" has the same horizontal time scale as the graph above but shows the time of YOUR CURRENT download. When you click the "DOWNLOAD" button, a vertical bar will appear in the far left side of this status bar. The height of this bar represents the net payoff of a successful download *if it finished at that time*. As you wait for the download, this bar moves from left to right and shrinks as your delay costs accumulate. If the download takes so long that the delay cost exceeds the 10 pt. value of the star, this bar drops below the middle line, indicating a negative payoff.

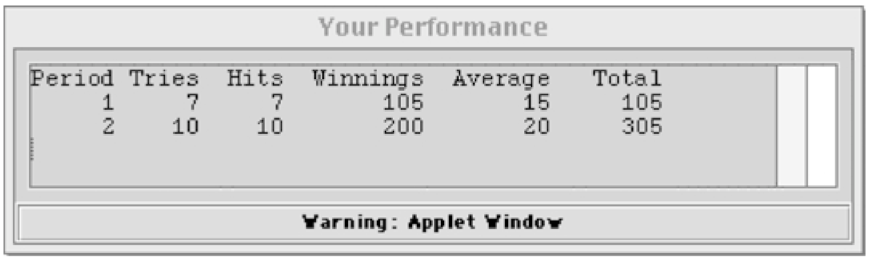
NOTE: Pushing the STOP button at any point will give you a lower payoff than the bar indicates by 10 points since you will not get the value of the star but still pay the delay cost.

In the window "Current Information" you will find out how much time passed on your last download attempt (Delay), what your earnings were for the last download attempt (Points), the number of your successful downloads in this period (Successful Downloads), your total amount of points for this period (Point), the time left in the current period (Time Left), and the time needed for a download in the last 10 seconds, averaged across all players (Group Average Delay).

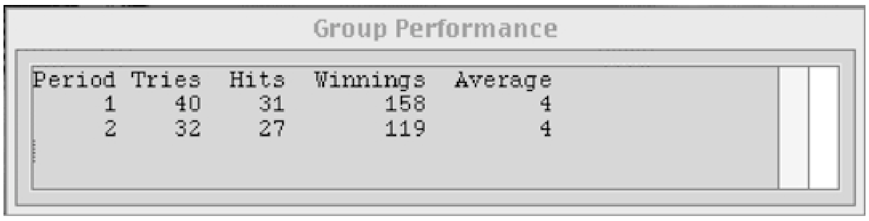
After the end of the first period two windows will appear on the right side of your screen. The top one displays information about your activity in the previous periods: number of attempted downloads (Tries), number of successful downloads (Hits), points (Winnings), your average points per try (Average), and a running total of your payoffs for all periods (Total). The bottom window shows the same statistics for the entire group. These windows will stay on your screen and will be updated at the end of each period.

IV. PAYMENT

The computer adds up your payoffs over all periods in the experiment. The last value in the 'Total' column in the 'Your Performance' window determines your



```
Period Tries  Hits  Winnings  Average  Total
     1     7     7       105       15    105
     2    10    10       200       20    305
```



```
Period Tries  Hits  Winnings  Average
     1    40    31       158        4
     2    32    27       119        4
```

payment at the end of the experiment. The money you will receive for each point will be announced and written on the board. After the experiment, the conductor will call you up individually to calculate your net earnings. You will sign a receipt and receive your cash payment of $5 for showing up, plus your net earnings.

V. FREQUENTLY ASKED QUESTIONS

Q: What happens if my net earnings are negative? Do I have to pay you?
A: No. To make sure that this never happens, you will be asked to leave the experiment if your total earnings start to become negative. In that case you would receive only the $5 show up fee.
Q: Is this some kind of psychology experiment with an agenda you haven't told us?
A: No. It is an economics experiment. If we do anything deceptive, or don't pay you cash as described, then you can complain to the campus Human Subjects Committee and we will be in serious trouble. These instructions are on the level and our interest is in seeing how people make decisions in certain situations.
Q: If I push STOP or RELOAD before a download is finished I get a negative payoff? Why?
A: Once you start a download, delay costs begin to accumulate. These costs are deducted from your total points even if you stop to download by clicking STOP or RELOAD.
Q: How is congestion determined?
A: Congestion is determined mainly by the number of download requests by you and other participants (humans and computer players). But there is also a random component so sometimes there is more or less background congestion.