The Potential for Cooperation among Web Agents

Cristina Bicchieri
Depts. of Philosophy and
Social and Decision Sciences
Carnegie Mellon University
cb36+@andrew.cmu.edu

Martha E. Pollack
Dept. of Computer Science and
Intelligent Systems Program
University of Pittsburgh
pollack@cs.pitt.edu

Carlo Rovelli
Dept. of Physics
University of Pittsburgh
rovelli+@pitt.edu

Abstract

In building intelligent network agents, computer scientists may employ a variety of different design strategies, and their design decisions can have a significant effect on the ultimate nature of network interactions. Some agent designs are "cooperative," and populations of agents based on them would be able to interact smoothly, effectively utilizing network resources. In contrast, other agent designs can lead to ineffective and wasteful competition for network resources, resulting in massive bottlenecks and unacceptable access delays. We focus here on a particular design question, the *multiple-access problem:* if an agent seeking a piece of information knows of several sites that have, or might have, that information, how many queries should it issue, and when? We provide a formal analysis that demonstrates the viability of cooperative responses to this question. We then discuss the limitations of this analysis and describe a simulation system we are building to study more general versions of the problem.

Introduction

An emerging vision of future massive information networks has them populated by intelligent electronic agents who are capable of understanding a user's specification of information goals, and deciding upon and carrying out the tasks necessary to achieve those goals. These agents will have knowledge of how to locate, access, and retrieve information from the network, thereby relieving users of the burden of having this knowledge themselves. Ideally, network agents will not be limited to tasks of pure information retrieval, but will also be capable of filtering email, scheduling meetings, and so on, and they will be capable of learning their users' preferences about how such tasks are to be performed. Simple intelligent network agents have already been built in early prototype form, e.g., (Lieberman 1995, Perkowitz & Etzioni 1995, Etzioni & Weld 1994, Kautz 1994).

In building intelligent network agents, computer scientists may employ a variety of different design strategies, and their design decisions can have a significant effect on the ultimate nature of network interactions. Some agent designs are "cooperative," and populations of agents based on them would be able to interact smoothly, effectively utilizing network resources. In contrast, other agent designs can lead to ineffective and wasteful competition for network resources, resulting in massive bottlenecks and unacceptable access delays. More generally, populations of agents on a network may be comprised of agents that employ different interaction strategies---for example, some percentage of them may be fully cooperative, another group may be partially cooperative, and the rest may be highly cooperative. Given this, we ask two questions:

1. What is the nature of network interactions for different such populations?

2. If agents are *adaptive*, i.e., capable of modifying their behavior in response to current network conditions, what will the population dynamic be? In particular, can adaptation of access strategy ever lead to cooperation, and, if so, under what circumstances?

In our ongoing research, we are addressing these questions using a variety of tools from artificial intelligence, distributed systems theory, and noncooperative and evolutionary game theory. Initially, we are focusing on the problem of information retrieval, in particular, on what we call the *multiple-access problem:* if an agent seeking a piece of information knows of several sites that have, or might have, that information, how many queries should it issue, and when? We have defined a set of information-access strategies that vary in the degree to which they are cooperative, and we are examining populations of agents that employ these strategies in varying proportions. The multiple-access problem is, of course, just one among many that must be addressed in the design of an intelligent network-agent; our focus on it is meant to be illustrative of the larger research enterprise, and to support the development of a model that can serve as the foundation for many related questions.

In this extended abstract, we briefly define the multiple-access problem, and sketch a formal analysis of one version of it that uses techniques from game theory and queuing theory. We then discuss the limitations of the formal analysis, and describe a simulation system we are building to study more general versions of the problem. This system will also serve as the basis of our experiments on adaptation and the possibility of emergent cooperation.

The Multiple-Access Problem

Problem Definition

As is well known to users of the World Wide Web, the same piece of information is frequently located at multiple sites. For example, someone seeking a local weather forecast can typically retrieve that information from more than a dozen different sites. Usually, a human Web user will visit one of those sites at a time; if she encounters a delay in accessing the first site she attempts to visit, she may then try a second, and so on. A similar situation arises when a Web user performs a net search, using a tool like Lycos or Netscape Search, which take as input keywords of interest, and return as output lists of sites that may contain relevant information. The usual strategy for the human user is to visit the listed sites on the one at a time, until the sought information is located.

In either scenario, the value of the information sought may be time-dependent; that is, there may be some penalty associated with a delay in getting the information. In some cases, e.g., for ordinary Web searches, the penalty may simply be the "opportunity cost," or annoyance experienced by the human user as she wastes time waiting for the information she needs. In other cases, the penalty can be much more severe: examples include the use of the Web by stock-traders to gain real-time information about financial markets, or its use by military commanders to gain real-time information about situation development.

This possibility of significant penalties for delayed information access raises an interesting question, which we call *the multiple-access problem*: If an agent seeking a piece of information knows of several sites that have, or might have, that information, how many queries should it issue, and when? Should several queries be issued simultaneously, rather than in the sequential fashion that is typical today? This question becomes particularly relevant once artificial agents are tasked with the job of information retrieval, because such agents could easily spawn several subprocesses (subagents) to access multiple sites simultaneously---a task that is more burdensome for the human user.

If there is no extrinsic cost associated with accessing a site, a plan to simultaneously issue multiple queries might appear to be a good one, increasing the likelihood of getting the information quickly, without incurring any cost. However, if multiple agents form such plans, a cost may in fact be incurred, as the result may be a highly overloaded system, in which all agents, on average, do worse.

A Simplified Case

We begin by considering a simplified version of the problem, in which there are exactly two agents and two sites that the agents both know to have the information sought. We assume that if both agents attempt to access the same site, there is an equal chance that either one will get there first. We assume further that both agents have the same, very simple payoff function: the utility of being the first to access one or both of the sites is h while the utility of being second to access both is l, under the constraint that $h > l$. Finally, we assume that there is no extrinsic cost associated to the access (i.e., all agents can submit information requests "for free"), that all hosts are deemed equally reliable, and that all this information is common knowledge.

With these assumptions, the decision that each agent must make can be modeled as a strategic-form game, as follows:[1]

	Access One	Access Both
Access One	$.75h + .25l$, $.75h + .25l$	$.5h + .5l$, h
Access Both	h, $.5h + .5l$	$.75h + .25l$, $.75h + .25l$

FIGURE 1. The Two Agent Case.

For example, if both agents adopt the strategy of accessing one site, then there is a 50% chance that they'll go to different sites (in which case each will receive a reward of h), and there is a 50% chance that they will go to the same site. In the latter case, for each agent, there is a 50% chance of getting there first (and being rewarded h), and a 50% chance of getting there second (and being reward l). Thus, each agent's expected utility in this case is $.5h + .5*.5h + .5*.5l = .75h + .25l$. Similar analyses can be given for the other cells.

What we see is that each player has a dominant strategy, to access both sites, and the result is a Nash equilibrium that is also Pareto-optimal. What this means, in brief, is that (i) each agent prefers to access both sites regardless of what the other does (dominant strategy); (ii) when an agent believes that the other agent will access both sites, then it also prefers to access both sites itself (Nash equilibrium); and (iii) there is no other combination of actions that leads to a better outcome for one agent without the other agent doing worse (Pareto-optimal). Thus, for the two-agent, two-data site case, under the assumptions we made above, there is no need for agents to explicitly cooperate with each

[1] For details of modeling games in strategic form, see (Osborne & Rubinstein 1994). Note that, as is standard in the strategic-form notation, the first payoff in every cell belongs to row-agent, and the second to the column-agent.

other: by simply choosing their dominant strategies, they achieve a result that is as good as if they were cooperating.

This fact changes, however, as we introduce more agents. For example, in Figure 2, we show the strategic-form game for three agents and two sites, under the assumption that being the first to access either site results in a payoff of *h*, being second at one site and second or third at another results in a payoff of *m*, and being third at all three sites results in a payoff of *l*, under the constraint that $h > m > l$. Note that Figure 2 lists the expected payoffs only for the row player. The expected payoffs for the other players are symmetrical.

Each agent still has a dominant strategy, to access both sites, and the situation in which everyone accesses both sites is the unique Nash equilibrium of the game. The equilibrium, however, is deficient, i.e., Pareto-suboptimal, since all would do much better were they to access only one site. We thus have a classical Prisoner's Dilemma. Agents would stand to gain were they capable of cooperating with each other and submitting queries to one site only. In fact, as the number of agents increases relative to the number of databases, the spread increases between the cooperative, Pareto-optimal outcome, in which all agents access one database, and the defective outcome, in which they all access both databases. The more agents there are in the system, the greater would be the gain from cooperation.

	Both Access One	One Accesses One	Both Access Both
Access One	$7/12\,h+1/3m + 1/12\,l$	$5/12\,h + 5/12m + 1/6\,l$	$1/3\,h+ 1/3\,m + 1/3\,l$
Access Both	$7/8\,h+ 1/8\,m$	$2/3\,h+ 1/3\,m$	$5/9\,h+ 3/9\,m + 1/9\,l$

FIGURE 2. The Three agent Case.

But how could such cooperation arise? In a one-shot encounter, there is no reason for an agent to be cooperative: if one expected the others to access only one site, the rational choice would be to access both. The situation changes, however, if agents interact repeatedly with each other. Traditional game-theoretic models show that, in case agents assess a sufficiently high probability of a future encounter, cooperative behavior can occur, since the gains from continuing cooperation far outweigh the gains of defection. Such models, however, are not fully applicable to the case at hand. First,, they assume that agents are not anonymous, but are able to identify one another, and thus to obtain "reputations", which can result in retaliation; we return to this point in the final section of the paper. Second, these models assume that agents are fully rational and have common knowledge of their mutual rationality, and that they are perfect calculators without any time or computational constraints. In fact, real agents (including Web agents) are boundedly rational: they have only limited knowledge of their environment and of other players, and they are computationally limited.

The Effect of Limited Knowledge

Perhaps surprisingly, the fact that Web agents have limited knowledge can be used to demonstrate that, at least in an idealized case, cooperation may occur among agents that are designed according to principles of rationality. The argument, which we sketch here, is presented in detail in (Bicchieri, Pollack, & Rovelli 1996), and hinges on agent's lack of complete knowledge about the state of their environment.

Specifically, assume that there are m sites and n agents, and that each agent submits a query with average frequency F. We assume that agents' queries are independent of each other, and that the frequency F is the same for all agents. Further, each database has a response time of ΔT, which is the same for all databases. At any time, a database can be either free or busy. If it is free and it receives a query, it becomes busy and stays busy for time ΔT, after which it sends an answer. If the database is busy while the query arrives, the query will wait in line until the database is free. Incoming queries will form a queue: for example, if there are five prior queries, the sixth query will be answered after $6\Delta T$. The maximum frequency at which a database replies is $1/\Delta T$.

As in the simplified case above, we consider only the two most "extreme" strategies, full cooperation (C), in which each agents sends one query to a randomly selected database and waits for a response, and defection (D), in which each agent send queries to all databases at once. (To simplify the analysis, we assume that all databases have the information sought.) We denote by n_c the number of agents using strategy C (the cooperators) , and by n_d the number of agents using strategy D (the defectors). Finally, we use T_c and T_d to represent the average waiting time for cooperators and defectors, respectively.

The average number of queries per period received by a database will be:

$$\frac{n_c F + n_d F m}{m}$$

We refer to this F_{DB}. Its inverse ($1/F_{DB}$) is the average waiting time between two queries in a database; we will call this W_{DB}. There are then three possible situations:

(i) $W_{DB} >> \Delta T$

In case (i), there is enough delay, on average, between the queries, to prevent queuing. Thus, there is no particular advantage to either cooperation or defection: $T_c = T_d = \Delta T$.

(ii) $W_{DB} << \Delta T$

In case (ii), the wait time is small, and the system is overloaded. As the inequality increases, the average waiting time for both types of agents goes to infinity.

(iii) $W_{DB} \approx \Delta T$

In case (iii), the average wait time between queries submitted is close to the response time for each database. In this case, defectors will achieve better performance than cooperators. This is because, even though we are assuming that queries are uniformly distributed in time and among the databases to which they are sent, in general, the queue lengths will not be identical at each database. The uniform distribution of queries does not imply that they are sent at constant time intervals from each other, or that precisely the same number of queries is sent in each unit of time. Rather, the number of queries generated in a given time interval will in general fluctuate from one time interval to the next, with the relative fluctuation going to zero only in the limit. Consequently, there will be variation among the queue lengths. Because defectors submit queries to multiple data sites (in fact, to all data sites), they have a higher probability of reaching a site with a small queue, and thus receiving a rapid response. In short, in case (iii), the average waiting time for cooperators is greater than for defectors, i.e. $T_c > T_d$.

Given this analysis, we can see that, if there is no cost associated with sending a query, then, at least under our current assumptions, agents should be designed to use a defector's strategy, since cooperation and defection are equivalent in cases (i) and (ii), while defecting pays in case (iii). Suppose, however, that query submission has a cost. Minimally, there is the opportunity cost of using resources that might be employed elsewhere; moreover, each answered query might cost the agent a fixed amount of money, and so might connection time. Even if these costs are rather insignificant, they are enough to tilt the scale in favor of cooperation. This is because, if queries have a cost, then in cases (i) and (ii), it will pay to be a cooperator rather than a defector: average waiting time is equivalent for both strategies, but the costs of defecting are higher. The agent must then determine the probability of being in each situation (i, ii, or iii). It follows from standard queuing theory that the probability of being in the transitional situation (iii) is extremely. (For details, see (Bicchieri, Pollack, & Rovelli 1995). For example, if the the agent's maximum waiting time---the amount of time it is willing to wait before "giving up"---is on the order of 1 second, and the database response time is 1 ms., then the probability of being in the transitional region is only 1 part in 2 million. Given these odds, it is reasonable to presume that the agent will seldom be in a type (iii) situation; consequently, it should be designed to use a cooperative strategy, which will achieve better performance in cases (i) and (ii).

Strategies for Information Access

The preceding section sketched a theoretical argument that cooperation could occur among Web agents. While it provides an important baseline analysis, it depends upon a number of very strong assumptions and simplifications. Amongst these is its focus on only two, rather extreme strategies for information access: a fully cooperative strategy (submit an information request to only one site), and a fully uncooperative strategy (immediately submit information requests to all potential sites). In fact, agents have a much larger set of possible access strategies, the space of which we can illustrate with a few examples, for which we provide pseudo-code below in Figure 3.

The most cooperative strategy involves accessing only a single site at a time. In one version of this strategy ("Unconditional Cooperation"), the agent picks a single site to which to issue a query, and then waits indefinitely for a response. In another, ("Impatient Cooperation") the agent cycles through the sites known to have the information in question, waiting for some fixed time period before retracting the query and issuing a new query to a different site. Somewhat less cooperative is the "Conditional Cooperation" strategy, in which the agent begins by issuing a query only to one site, but, if it fails to receive a response within some fixed time, issues another query to a second site without retracting the first query, and so on. As time passes, this agent will have queries outstanding at more and more sites. A somewhat more extreme version of this approach is "Grim Cooperation," in which the agent begins by being cooperative, issuing a query only to one site, but if an answer is not received within a fixed time period, it immediately issues queries to all remaining known sites. The least cooperative strategy is "Defection," which involves immediately issuing requests to all sites known to have the required information.

Given a set of such strategies, one would like to be able to analyze the performance of populations agents using them. However, a clean formal analysis, using the tools of game theory, is quite difficult. In part, this is due to the complex behavior that results from the larger space of strategies. In part, it is due to other limitations of game-

theoretic tools that preclude their application to the general Web agent problem.

Unconditional Cooperation
i := pick-db;
issue-query-to (i);
while data not retrieved
wait.
	Impatient Cooperation
	i := pick-db;
	issue-query-to (i);
	t := 0;
	while data not retrieved
	begin
	while t < T
	if data retrieved then
	exit
	else t : = t + 1;
	retract-query-from (i);
	i := pick-db;
	issue-query-to (i);
	t := 0;
	end.

Conditional Cooperation
i := pick-db;
issue-query-to (i);
t : = 0;
while data not retrieved
begin
while t < T
if data retrieved then
exit
else t := t + 1;
if not all db's queried then
begin
i := pick-db;
issue-query-to (i);
end;
t := 0;
end.
	Grim Cooperation
	i := pick-db;
	issue-query-to (i);
	t := 0;
	while t < T
	if data retrieved then exit
	else t : = t + 1;
	for all other db's j
	issue-query-to (j);
	while data not retrieved
	wait.

Defection for all db's j issue-query-to (j); while data not retrieved wait.	

FIGURE 3. Access Strategies

These limitations include difficulties in:

- handling asynchronous interactions. Traditional game theory assumes sequential, synchronized interactions, but no synchronization can be assumed to hold among network agents.

- handling situations in which the sets of agents involved in sequential interactions overlap with each other. Traditional game theory can handle the case of a new agent entering into a game only if the agents involved come with reputations from past play. However, in the context of network agents, different sets of agents will be seeking information at different times, and the community will be too large to ensure readily available "reputations".

- having agents infer the strategies being used by other agents. In the traditional game theory literature, agents know, at the end of each interaction, what action the other agents have taken. In the network agent setting, this is not the case: agents have no privileged access to the actions of other agents. They must infer, on the basis of what they can observe, what other agents are doing. For example, an agent may only know that it has an average access delay that is greater than what is expected, and may, on the basis of this, conclude that other agents are failing to behave cooperatively.

- integrating the problem of deciding among a set of options for action with the problem of determining what those options are in the first place. Traditional game theory assumes that the "game" is specified, i.e., that the agents know what their choices for action are. Within AI, the field of planning has focused on the question of how an automated agent can determine what actions can lead to its goals. A merging of these two efforts in needed.

Given these factors, we are taking a two-pronged approach to the task of modeling Web agents more realistically. First, we are working to extend game-theoretic tools, to remove some of the limitations just mentioned. Second, we are developing a simulator that can serve as the platform for Web population studies, to determine whether cooperation is in fact feasible under more realistic conditions than those assumed in the formal analysis above.

Adaptive Strategies

Even if we are successful in modifying standard game-theoretic tools to meet the challenges outlined above, this will constitute only a first step. This is because, as we noted earlier in the paper, these presuppose that agents repeatedly interact with the same parties, can recognize and monitor each other's behavior, and can thus effectively employ retaliation when needed. This is not the case in large, anonymous groups, for example, collections of network agents. When the number of agents is large, defection may go undetected; even if it may be detected, it might prove too costly (and ultimately useless) for an agent to punish the defector, since the probability of future encounters is vanishingly small.

We are thus also examining the relevance of the work on evolutionary models to the network agent problem (Axelrod 1984, Smith 1982). In such models, agents may not only incorporate strategies for interaction, but also may have a meta-rule for switching strategies: such meta-rules constitute adaptive algorithms. Among a population of adaptive agents, strategies may be viewed as being in competition, in which only the "fittest" survive. It can be shown that, depending on the initial environment, strategies that lead to cooperation in Prisoner's Dilemma-type games outperform strategies that don't. An appealing feature of such models is that, in an evolutionary context, agents are modeled as limited reasoners. In this framework, a strategy can be seen as a recipe for converting information acquired during past interactions into actions to be taken in the future. Moreover, these strategies require very limited computational resources and are therefore quite appropriate for resource-bounded agents such as network agents.

Well-known evolutionary models such as Axelrod's have only studied the evolutionary course of strategies that are matched pairwise in round-robin tournaments. In the case of network agents, however, we must consider encounters among more than two, and potentially many, agents; this, again, is where our simulator will prove useful. It then matters whether the population is fixed or not, and whether agents have the possibility of repeated encounters with at least some small subsets of the population. Bicchieri and Rovelli (1995) have shown that, if the population is fixed, cooperation can emerge as a stable strategy, provided that defection creates small but cumulative overall costs and that there exists a (very) small but stable number of conditional cooperators in the population. This is a step in the right direction, but one must go even further for the network agent application, in which the population is not fixed: agents continually enter and exit the environment.

One way to approach this situation is to view the agents as clustered into local "neighborhoods," in which they have repeated interactions with each other for sufficiently long periods. Examples might include users of a LAN such as co-workers within an individual office or department. Although it is unlikely that two such users (or their agents) would typically query the same end database at the same time, it is much more likely that they frequently submit queries that must pass through the same intermediate servers, resulting again in a need for cooperation.

The expectation is that adaptive agents involved in such local interactions are likely to develop cooperative behavior. Moreover, since they are adaptive and will therefore tend to keep using a strategy that did well in the past, and change it otherwise---cooperative behavior may spread from such local "neighborhoods" to much larger groups, provided that the number of cooperators entering the bigger group is large enough. This, however, is currently just a hypothesis. The focus of our ongoing efforts is to explore whether in fact the Web agent's environment will be one in which cooperative behavior can emerge.

Acknowledgments. Principal support for this work has come from the Office of Naval Research (Contract N00014-95-1-1161). Pollack has received additional support from the Rome Laboratory (RL) of the Air Force Material Command and the Advanced Research Projects Agency (Contract F30602-93-C-0038), and from an NSF Young Investigator's Award (IRI-9258392).

References

R. Axelrod, *The Evolution of Cooperation.* Basic Books, New York, 1984.

C. Bicchieri, M. E. Pollack, and C. Rovelli, "Cooperation among Network Agents," in preparation.

C. Bicchieri and C. Rovelli, "Evolution and Revolution: The Dynamics of Corruption," *Rationality and Society*, 7, 1995.

O. Etzioni and D. Weld, "A Softbot-based Interface to the Internet," *CACM* 37(7):72-76, 1994.

H. Kautz et al., "An Experiment in the Design of Software Agents," in *Proceedings of AAAI-94,* pp. 438-443.

H. Lieberman, "Letizia: An Agent that Assists Web Browsing," in *Proceedings of IJCAI-95,* pp. 924-929.

M. J. Osborne and A. Rubinstein, *A Course in Game Theory*, MIT Press, Cambridge, 1994.

M. Perkowitz and O. Etzioni, "Category Translation: Learning to Understand Information on the Internet," in *Proceedings of IJCAI-95,* pp. 930-936.

J. M. Smith, *Evolution and the Theory of Games*. Cambridge University Press, Cambridge, 1982.