Agent Amplified Communication

Henry Kautz, Al Milewski, and Bart Selman
AT&T Bell Laboratories
Murray Hill, NJ 07974
{kautz, aem, selman}@research.att.com

Abstract

We propose an agent-based framework for assisting and simplifying person-to-person communication for information gathering tasks. As an example, we focus on expertise location within a large organization. In our approach, the informal person-to-person networks that exist within an organization are used to "referral chain" requests for expertise. User-agents are employed to automate the referral chaining process. We also present simulation results demonstrating the effectiveness of our approach.

Introduction

There are basically two ways of finding something out by using a computer: "ask a program" and "ask a person".

The first covers all ways of accessing information stored online, including the use of traditional database programs; file indexing and retrieval programs such as glimpse (Manber and Wu 1994) or Apple's AppleSearch; news filtering programs such as Hoover (SandPoint Corp.); and even more simply, the use of tools such as ftp, awk, and text editors to retrieve and view files.

The second, "ask a person", covers ways that a computer can be used as a communication medium between people. Currently the prime examples are electronic mail, including both personal e-mail and mailing lists, and bulletin boards and newsgroups. The growing integration of computers and telephones allows us to also view telephony as a computer-based communication medium. Simple examples of such integration are telephone address book programs that run on a personal or pocket computer and dial numbers for you; more sophisticated is the explosion in the use of computer-based FAX. Today it is hard to even buy a modem that does not have FAX capability, and by far the heaviest use of FAX is for person-to-person communication.

There are inherent problems with both general approaches to obtaining information. It has often been noted that as the world of online information sources expands, the "ask a program" approach suffers from the problem of knowing where to look. For example, the Mosaic system overcomes many of the technical problems in accessing a wide variety of information on the Internet, by automatically handling the low-level details of different communication protocols. It is easy and entertaining to browse through an enormous hypermedia space. However, finding an answer to a specific question using Mosaic tends to be slow and frustrating, and often results in failure. One response to this problem has been the attempt to design systems that incorporate knowledge about the location of information (Etzioni and Weld 1994; Kirk *et al.* 1995; Knoblock *et al.* 1994; Maes 1993). However, a deeper problem remains, that no solution based solely on building a better search-engine can address. This is the fact that much valuable information is simply not online, but only exists in people's heads. Furthermore, there are economic, social, and political reasons that much valuable information will never be made publicly accessible on the Internet or any other network. Indeed, part of the value of a piece of information resides in the degree to which it is not easily accessible.

This is perhaps most obvious in relationship to proprietary corporate information. For instance, if I am involved in trading General Motors stock, I may be vitally interested in knowing the specifications of the cars to be introduced next year. That such information exists is certain – indeed, factories are already being set up to produce the vehicles – but I am certainly not going to be able to find this information in any database to which I have access.

For a more mundane example (at least one of less concern to the SEC), suppose I need to have my house painted, and want to know if Acme Painters Inc. does good work. It is highly unlikely that I am going to be able to access a database of "reliable housepainters". Conceivably the Better Business Bureau might offer a service that could tell me whether many people have actually taken Acme to court, but that would hardly be all that I would want to know. Any recommendations offered by such a public service would have to be either advertisements or sufficiently innocuous to avoid legal entanglements. An even more telling case would be where I am trying to decide whether or not to hire a certain professor John Smith to do some consulting for me, and I want to know whether or not Smith knows what he is talking about. It is certainly not the case that I will be able to access a database of academics, with entries such as "solid, careful thinker" or "full of hot air".

Thus, many important kinds of information can only be obtained by what we called "ask a person". As with the "ask a program" paradigm, there is an initial technical problem of establishing a communication link that is yielding to current technology. Bridges exist between most major e-mail systems, the Internet domain naming standard is becoming universally adopted, and eventually global directory services (very roughly approximated by the current "netfind" program) will make it easy to determine the electronic address for anyone in the world. Looking further into the future we see wireless personal communicators replacing stationary telephone sets and computer terminals, so that all of electronic mail, voice mail, and ordinary telephone calls can really be routed directly to a person, rather than merely to a location that the person is likely to frequent.

But in this rosy scenario the thorny problem of determining how to ask the *right* person remains. Quite often someone I know (or someone with whom I have a potential professional or social relationship) has the information that I need, but I am not sure who they are. At first e-mail seems to offer to offer a solution to this problem: just mail the question to everyone who I think *might* know the answer. It is easy to add recipients to a message, and easier still to send mail to an alias that expands to a large number of potential candidates. But in this context the phrase "potential candidate" might better be replaced by "potential victim", because such wide-area broadcasting of electronic mail quickly becomes obnoxious. In getting one person to help me, I annoy hundreds or even thousands of others. For example, using one alias I can easily send mail to everyone who works at AT&T, that is, more than 100,000 people (and from time to time someone does just that, to his or her everlasting regret). If I am persistent in this kind of activity, people will soon ignore my requests; in their view, I am lumped together with door to door salesmen and telephone solicitors. Ultimately mail I send may be electronically filtered out by my intended recipients, and I may face even more serious consequences. A relatively mild rebuke that was recently sent to members of our lab concerning our "news" alias ran as follows:

```
From XXXXX Thu Aug 25 15:57:18 1994

To maintain 'news' as a useful channel
for our professional interests, it
is necessary to avoid other kinds of
information.

Netnews is available for selling cars,
giving away kittens and puppies, and
many other purposes.

For many years, it has been clearly
if tacitly understood that 'news'
should only be used for items of
professional interest.  Please --
let's not litter this unique channel
with unrelated items.
```

So, what about netnews? Posting to netnews does eliminate the annoyance factor, because only people who actively want to read the messages do so. Unfortunately the people I truly wish to reach, those who have the valuable information that I need, are the least likely to read netnews. Many would agree that as access to and the volume of netnews has increased over the past years, the "quality level" of most newsgroups (never that high to begin with) has declined. Informed, busy people simply drop out of the electronic community, so that most groups become forums for ill-informed opinions, unanswered pleas, downright misinformation, and various exhibitions of social pathology.

The current tools for "ask a program" and "ask a person" are largely disconnected at the top level, despite the fact that they rely on a shared electronic infrastructure. We believe that systems that integrate the two paradigms can provide solutions to the problems inherent in each. We are designing and building systems that use software agents to assist and simplify person-to-person communication for information gathering tasks; we call this approach *agent amplified communication.*

Expertise Location

In a previous paper (Kautz *et al.* 1994), we described our view of software agents and the particular agent platform we had built. We take agents to be programs that assist users in routine, communication intensive activities, such as locating information, scheduling meetings, coordinating paper reviews, and so on. A user delegates a task to an agent, which can then engage in many transactions with other agents and other people. Delegation thus reduces the total communication load on the user. As a philosophical point, we believe that agent systems should blend transparently into normal work environments and existing infrastructure, and take advantage of (rather than trying to replace) the networks of formal and informal systems and social relations that exist in an organization.

The system we built used two basic classes of agents: task specific agents, called "taskbots", and personal agents for each user, called "userbots". Our initial task specific agent was used for scheduling meetings with visitors to our lab, and was thus named the "visitorbot". A host would merely need to provide the visitorbot with a talk abstract and the times that a visitor was available for meetings, and the visitorbot would carry out all the steps necessary to set up a series of meetings (i.e., sending out a talk announcement, obtaining preferred meeting times from interested parties, and generating and distributing a schedule for the day). The userbots provide a graphical, customizable interface to the taskbots, as well as a repository for information that is private to the user. For example, a user could tell his userbot his preferred meeting times, and the userbot would then transmit this information to the visitorbot.

We are now designing a new generation of taskbots and userbots for information-gathering tasks of the kind described above. The specific task we are working on is expertise

location. In any large organization, determining who is an expert on a particular topic is a crucial problem. The need for expertise location ranges from informal situations, such as where I might need to find an expert on LaTex macros to help fix a typesetting problem in a paper I'm writing, to formal construction of project teams to meet business needs. The range of expertise specifications may range from the generic ("who knows about logic programming?") to the highly specific ("who knows how to modify the interrupt vector handling microcode in the reboot module of the XZY999 processor?").

Online directories of expertise rarely exist, and when they do, the information that they contain is certain to be out of date and incomplete. In fact, expertise needs are potentially so specific that it is simply impossible to determine a comprehensive set of categories in advance. Expertise location is therefore generally an "ask a person" task, with the all the problems associated with that approach outlined above.

Let us consider for a moment how expertise location actually works when it is successful. In a typical case I contact a small set of colleagues whom I think might be familiar with the topic. Because each person knows me personally, they are quite likely to respond. Usually none of them is exactly the person I want; however, they can refer me to someone *they* know who might be. After following a chain of referrals a few layers deep I finally find the person I want.

Note that in this successful scenario I needed to walk a fine line between contacting too few people (and thus not finding the true expert) and contacting too many (and eventually making a pest of myself). Even in the end I might wonder if I might not have found even a better expert if only I could have cast the net a bit wider. I may have had difficulty bringing to mind those people I do know personally who have some expertise in the desired area. If only all of my colleagues employed endlessly patient assistants that I could have contacted initially, who would have known something about their bosses' areas of expertise, and who could have answered my initial queries without disturbing everyone...

Now let us consider how software agents could be used to augment the expert location process. Each person's userbot would create a model of that person's areas of interest. This model would be created automatically by using information retrieval techniques (such as inverted indexes) on all the documents created and received by the user. The user model could be quite large and detailed, and would be private to the user, that is, not stored in a central database. The userbot would also create a much more coarse-grained model of my contacts by applying similar techniques to all the electronic mail that I exchange with each person.

When I have an expertise location need, I present the problem to my userbot as an unstructured text description. Again using IR techniques, my userbot selects a medium-to-large set of my contacts to whom the query may be relevant. It then broadcasts the query, not to the people themselves, but to their userbots. Upon receipt of the question, each userbot checks if it's owner's user model does indeed provide a good match. If there is a good match, the userbot presents my request to it's owner. If the owner's model does not match, but the model of one of the owner's contacts does, then the userbot can ask the owner if it can provide a referral. Finally, if there is no match at all, the query is silently logged and deleted. A great deal of flexibility can be built into each userbot, depending upon it's owner's preferences. For example, I might allow automatic referrals to be given to requests that come from my closest colleagues.

This system provides several benefits over the netnews and the "send personal e-mail to everyone" approaches described above. First, it is largely passive on the part of the recipients – they do not need to be reading netnews and wading through dozens of articles. Second, queries are broadcast in a focused manner to those who are at least somewhat likely to find them of interest. Third, users are shielded from seeing a large number of completely irrelevant messages; each userbot may process dozens of messages for every one the user sees. Finally, messages that a user does see do not come from "out of the blue", but rather are tagged with a chain of referrals from colleague to colleague.

One reason to believe that the system just described would be useful in practice is that it basically models the manner in which expertise location actually works now, while allowing more people to be contacted without causing disruption and disturbance (Krackhardt and Hanson 1993; Grosser 1990).

Implementation of an Expertise Locator

An initial version the expertise location has been implemented by extending the user-agents as developed in our earlier "visitorbot" project (Kautz *et al.* 1994). Each user-agent has access to the following kinds of database files, each of which is specific to and owned by the individual user. It is important to note that we do not assume that these files can be directly accessed by anyone other than the user and the user's agent.

1. A user-contacts file containing a list of some of the user's colleagues, and for each a list of keywords describing their areas of expertise.

2. A file containing all of the email that the user has sent and received for a substantial period of time: typically, the past year or several years.

3. A indexed-email file that stores for each word that appears in any email message, a list of the messages that contain that word. This kind of file, called an "inverted index", can be generated using standard information retrieval algorithms, such as those described in Salton *et al.* (1989).

4. A email-record file that stores for each message number the identifier of the sender of the message (if other than the user) or the identifier of the recipient of the message (if sent by the user).

5. A user-profile file containing a list of keywords that describe some of the user's own areas of expertise.

6. A user-index file containing an inverted index of all of the

other files in the user's directory. That is, for every word that appears in any file the user has stored on the computer, this file contains a list of the names of the files containing that word. This kind of very large inverted index can be quickly created and searched by the program "glimpse" (Manber and Wu 1994).

A user begins the process of locating an expert in a topic by clicking on the user-agent window and typing a phrase that describes the general kind of request (such as, "I need to locate an expert"; another phrase may initiate other kinds of agent-assisted activities, such as the visitor-scheduling function mentioned above). The user agent then prompts the user for a phrase describing the area of expertise. Once this is done, the user agent generates and presents for approval a list of suggested candidates for receiving the request.

The list of candidates is generated by combining names from two sources. First, names are added that appear in the user-contacts file, such that the words that appear in the phrase describing the expertise request appear in the list of keywords associated with the name. Second, the email records are skimmed to determine for each potential contact person how many of their messages mention one or more of the keywords. The result is a list of pairs of "person name" and "number of messages". This list is sorted according to "number of messages". The 20 names with the highest number of messages in this list are then added to the list of candidates.

After being approved, the expertise location request is formatted as an email message containing special fields that indicate that it should be read and processed by the recipients' user agents, rather than the users' themselves. Each recipient user agent notes the "Msgtype" field, removes the message from the incoming mail stream, and processes it as follows:

First, the words in the expertise description phrase contained in the message are matched against the recipient's user-profile file. If the words appear in that file, then the user-agent assumes that this request is appropriate for the recipient to see.

If the words in the phrase do not match against the contents of the user-profile file, the user-agent uses the user-index file to match the phrase against the contents of all of the recipient's files. This matching can be efficiently performed using the program "glimpse" mentioned above. If the number of matches is greater then a threshold number (e.g., more than 10 matches), the recipient's user agent guesses that this request is likely to be appropriate for the recipient.

If the recipient's user agent thus determines in either way that the message is appropriate, it forwards to message to its owner. The recipient is then given the option of (i) responding affirmatively back to the sender; (ii) responding negatively back to the sender; or (iii) referring the request to someone else. In this final option is selected, the recipient's user agent creates a list of candidate recipients as described above and the process is repeated.

R	A	av. final dist.	av. number messages	success rate
1.00	1.00	0.0	12.1	100.0
1.00	0.70	0.0	42.6	99.7
1.00	0.50	0.0	109.8	99.1
1.00	0.30	0.1	377.2	94.6
1.00	0.10	0.6	1225.2	52.5
1.00	0.00	1.2	1974.6	8.5
0.90	0.90	0.1	21.3	95.2
0.90	0.70	0.2	43.8	92.0
0.90	0.50	0.3	92.1	83.1
0.90	0.30	0.7	204.9	53.2
0.90	0.10	1.5	316.3	13.3
0.70	0.90	0.7	16.0	60.3
0.70	0.70	1.1	22.3	39.0
0.70	0.50	1.7	27.5	19.6
0.70	0.30	2.2	28.3	6.8
0.70	0.10	2.5	31.9	1.2
0.50	0.90	1.4	9.5	26.6
0.50	0.70	1.9	10.5	11.1
0.50	0.50	2.3	10.8	5.2
0.50	0.30	2.6	10.7	3.1
0.50	0.10	2.9	11.1	0.2
0.10	1.00	2.2	3.3	1.3
0.10	0.70	2.7	3.5	0.3
0.10	0.50	2.9	3.5	1.1
0.10	0.10	3.2	3.5	0.1

Table 1: Simulation results for a 100,000 node network.

We are currently experimenting with email records of various volunteers in our center to determine how accurate the various expertise matches and referral suggestions are when based on email records and collections of user files. It is clear, of course, that the automatically generated matches will be less accurate than when people are asked directly. In the next section, we consider the effect of this loss of accuracy on the referral chaining process.

Measuring Effectiveness of Amplified Communication

A key question concerning agent amplified communication is whether it will actually improve upon the traditional person-to-person referral chaining process. It is clear that agents can handle larger numbers of messages, which may increase the chance of finding the desired information. On the other hand, as noted above, referrals suggested by user agents will generally be less accurate than those provided by people. This decreases the effectiveness of the referral process, and could in fact render the process ineffective. In order to study this issue, we ran a series of simulations of agent amplified communication.

In our simulation, we consider a communication network consisting of a graph, where each node represents a person with his or her user agent, and each link models a personal contact. In the referral chaining process, messages are sent along the links from node to node. Some of the nodes are

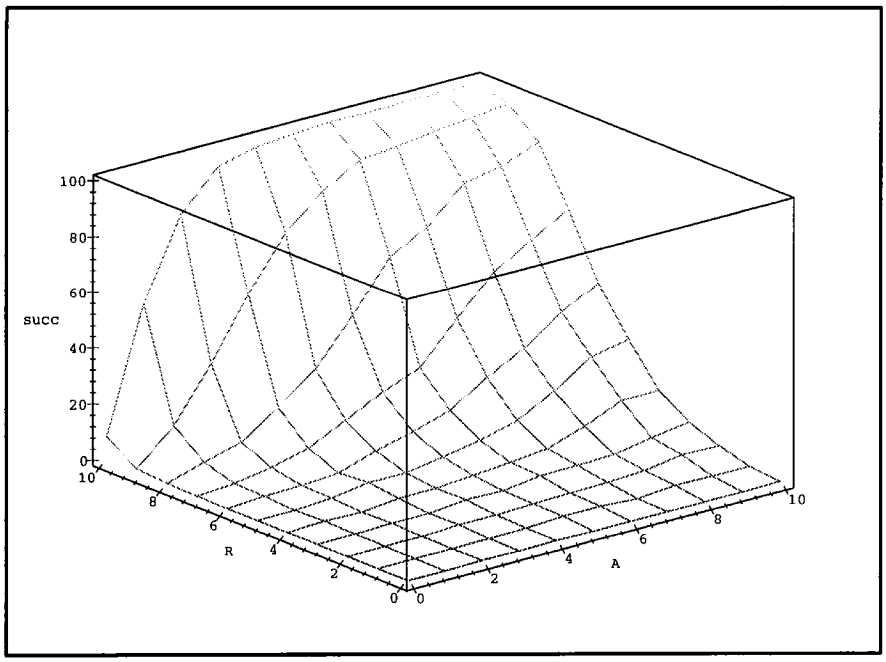

Fig. 1: Success rate as a function of responsiveness and referral accuracy. (Data from Table 1.)

designated "expert nodes", which means that the sought-after information resides there. In each simulation cycle, we randomly designate a node to be the "requester"; the referral chaining process starts at that node. The goal is to find a series of links that leads from the requester node to an expert node. We now introduce several parameters that further characterize the referral chaining process.

First, we have to model the fact that the further removed a node is from an expert node, the less accurate the referral will be. Why we expect the accuracy of referral to decrease is best illustrated with an example. Assume that one of your colleagues has the requested expertise. In that case, there is a good chance that you know this and can provide the correct referral. However, if you don't know the expert personally, but know of someone who knows him or her, then you may still be able to refer to the correct person (in this case, your colleague who knows the expert), but you'll probably be less likely to give the correct referral than if you had known the expert personally. In general, the more steps away from the expert, the less likely it is that you will provide a referral in the right direction. To model this effect, we introduce an *accuracy of referral factor* A (a number between 0.0 and 1.0). If a node is d steps away from the nearest expert node, it will refer in the direction of that node with a probability $p(A,d) = A^{\alpha d}$, where α is a fixed scaling factor. With probability of $1 - p(A,d)$, the request will be referred to a random neighbor (*i.e.*, an inaccurate referral).

Similarly, we model the fact that the further removed a node is from the requester node, the less likely the node will respond. This aspect is modeled using a *responsiveness factor* R (again between 1.0 and 0.0). If a node is d links removed from the originator of the request, its probability of responding will be $p(R,d) = R^{\beta d}$, where β is a scaling factor. Finally, we use a branching factor B to represent the average number of neighbors that will be contacted or referred to by a node during the referral chaining process.

R	A	av. final dist.	av. number messages	success rate
0.90	0.70	0.1	90.1	97.1
0.90	0.50	0.1	200.4	96.4
0.90	0.40	0.1	307.4	93.3
0.90	0.30	0.2	558.3	86.7
0.90	0.20	0.4	955.8	68.9
0.90	0.10	0.7	1466.4	43.9
0.70	0.90	0.4	32.4	74.5
0.70	0.70	0.7	52.4	57.1
0.70	0.50	1.2	72.2	32.3
0.70	0.30	1.7	87.0	14.1
0.50	0.90	1.2	17.6	29.4
0.50	0.70	1.7	20.0	16.8
0.50	0.50	2.1	21.4	6.6

Table 2: Simulation results for the network from Table 1, but with $B = 4$.

Table 1 gives the results of our first simulation experiment. The network consists of a randomly generated graph with 100,000 nodes. Each node is linked on average to 20 other nodes. Five randomly chosen nodes were marked as expert

R	A	av. final dist.	av. number messages	success rate
0.90	0.90	0.1	10.1	95.8
0.90	0.50	0.1	39.6	93.4
0.90	0.30	0.2	99.2	83.3
0.90	0.10	0.8	246.9	37.8
0.70	0.90	0.3	9.7	78.4
0.70	0.50	0.9	20.3	42.3
0.70	0.30	1.4	26.3	20.1
0.70	0.10	1.9	30.7	4.6
0.50	0.90	0.8	7.2	45.7
0.50	0.70	1.1	9.0	30.4
0.50	0.50	1.5	9.7	17.2
0.50	0.30	1.9	10.7	8.8
0.50	0.10	2.2	10.9	1.3
0.30	0.90	1.2	5.0	21.9
0.30	0.70	1.6	5.6	11.5
0.30	0.50	1.8	5.8	7.9
0.30	0.10	2.4	5.9	0.4

Table 3: Simulation results for a network as in Table 1, but with an average of 50 neighbors per node.

R	A	av. final dist.	av. number messages	success rate
0.90	0.90	0.1	30.1	94.7
0.90	0.70	0.2	77.3	89.1
0.90	0.50	0.7	170.8	60.7
0.90	0.30	1.4	281.9	22.9
0.90	0.10	2.2	329.3	3.2
0.70	0.90	1.0	21.5	46.7
0.70	0.70	1.6	26.1	23.4
0.70	0.50	2.3	29.6	7.5
0.70	0.30	2.8	31.4	1.8
0.70	0.10	3.1	33.3	0.3
0.50	0.90	1.9	9.9	11.9
0.50	0.70	2.5	10.8	5.5
0.50	0.50	3.0	10.6	1.8
0.50	0.30	3.3	10.9	0.2
0.30	0.90	2.5	6.0	2.9

Table 4: Simulation results for the network from Table 1, but with only 1 expert node in the net.

nodes. The average branching B, while referral chaining, is fixed at 3. The scaling factors α and β are fixed at 0.5. The table shows results for a range of values of R and A.

The final column in Table 1 gives the success rate of the referral chaining process, *i.e.,* the chance that an expert node is found for a given request. (The values in the table are based on an average over 1,000 information requests for each parameter setting.) We also included the average number of messages sent when processing a request. Note that because of the decay in responsiveness, at some distance from the requester node, the referral process will die out by itself. So, the processing of a request ends when either the chain dies out or an expert source node is reached. The third column shows how close (on average) a request came to an expert node. The distance is in terms of number of links. (This average includes the successful referral runs, where the final distance is zero.) On average, in the network under consideration here, the length of the *shortest* chain between a random node and the nearest expert nodes is about four links.

Our simulation results reveal several surprising aspects of the referral chaining process. Let us first consider some of the extreme cases. With $R = 1.0$ and $A = 1.0$, *i.e.,* total responsiveness and perfect referral, we have a 100% success rate, and, on average, it takes about 12 messages to process a request. Now, if we reduce the accuracy of referral to 0.0 (*i.e.,* nodes will simply refer to random neighbors), our success rate drops to 8.5%,[1] and it takes almost 2000 messages per request. There is a reasonably intuitive explanation for the large number of messages and the low success rate. Without any referral information, the search basically proceeds in a random manner. With 100,000 nodes, and 5 expert nodes, we have 1 expert per 20,000 nodes. In a random search, a request would visit on average about 20,000 nodes to locate an expert. (For simplicity, we assume a completely connected graph where the request travels randomly from node to node, and may visit a node more than once.) Given that the search only involves 2,000 message, the measured value of 8.5% is intuitively plausible.[2] Comparing these numbers with perfect information chaining, where we get a 100% success rate using only 12 messages, reveals the power of referral chaining.

Of course, in practice we have neither perfect referral nor complete responsiveness. Also, in our agent amplified communication approach, we are specifically interested in the question of to what extent the referral chaining process degrades with a decrease in referral accuracy. Let's consider this issue by looking at the results for $R = 0.9$ and $A = 0.3$, *i.e.,* a slightly lower responsiveness but drastically reduced referral accuracy. Table 1 shows that we still have a success rate of over 50%, and it takes about 200 messages per request. These numbers suggest that with a reduced referral accuracy, one can still obtain effective referral chaining, albeit at the cost of an increase in the number of messages. This should not pose a problem in our agent amplified communication model, where many messages will be processed automatically. Also, the 200 messages should be compared to the approximately 10,000 messages that would be required in a search without any referral information to reach the same success rate (see argument above). So, even with limited referral information, the referral chaining process can be surprisingly good, which suggests that an agent amplified approach can be very effective.

[1] With maximum responsiveness, the referral process would only halt when an expert node is found, giving a success rate of 100%. However, in our simulation we used a maximum referral chain length of 10. This limit was only reached when R was set to 1.0.

[2] Note that our argument is only meant to provide some intuition behind the numbers. A rigorous analysis is much more involved. See, for example, Liestmann (1994).

Let us also briefly consider a setting of the parameters that would model more closely direct person-to-person referral chaining. In that case, we would expect a lower responsiveness, but a much higher referral accuracy. For example, consider $R = 0.5$ and $A = 0.9$. We obtain a success rate of around 27%, with about 10 messages per request. So, with relatively accurate, but not perfect, referral, we still need only a few messages to get reasonably successful referral chaining. This is consistent with the observed effectiveness of the referral process in real organizations (Krackhardt and Hanson 1993).

In general, our simulation experiments suggest that a decrease in referral accuracy can be compensated for by having a somewhat higher responsiveness. The plot in Fig. 1 gives the success rate as a function of the responsiveness and referral accuracy. The figure nicely confirms a fairly smooth tradeoff between the two parameters.

Finally, the reader might wonder about the effect of the particular settings of the various network parameters in our simulation. Tables 2, 3, and 4 give the results of some of our other simulations. We only included the most informative settings of R and A. Each experiment was based on a modification of a single parameter used in the simulation experiment for Table 1. (See the caption of each table.) The results in these tables show again that one can reliably trade referral accuracy for responsiveness, and vice versa. We are still conducting a more exhaustive exploration of the parameter space.

Conclusions

We studied the use of agents in assisting and simplifying person-to-person communication for information gathering tasks. As an example, we considered the task of expertise location in a large organization. We discussed how user-agents can enhance the referral chaining process for expertise location. In our approach, agents gather information about the expertise of their owner and their owner's contacts. This information is then used in the referral chaining process to filter messages and to automate request referrals. We also presented simulation results of the referral chaining process. Our experiments suggest that an agent amplified communication strategy can be surprisingly effective.

References

Coen, M. 1994. SodaBottle. *M.Sc. Thesis, MIT AI Lab.*

Etzioni, O., and Weld, D. S. 1994. A softbot-based interface to the internet. *Communications of the ACM*, July 1994.

Grosser, D. 1990. Human communication. *Annual Review of Info. Sci. and Techn.*

Kautz, H.; Selman, B.; and Coen, M. 1994. Bottom-up design of software agents. *CACM*, July 1994.

Kirk, T.; Levy, A.; and Srivastava, D. 1995. The information manifold project. In *Proceedings AAAI-95 Spring Symposium on Information Gathering in Distributed Environments.*

Knoblock, C.; Arens, Y.; and Hsu, C.-N. 1994. Cooperating agents for information retrieval. In *Proceedings of the Second International Conference on Cooperative Information Systems.*

Krackhardt, D., and Hanson, J. 1993. Informal networks: The company behind the chart. *Harvard Business Review.*

Liestmann. 1990. Broadcasting and gossiping. *Discrete Applied Mathematics.*

Maes, P., ed. 1993. *Designing Automomous Agents.* MIT/Elsevier.

Manber, U., and Wu, S. 1994. Glimpse: A tool to search through entire file systems. In *Usenix Winter 1994 Technical Conference*, 23–32.

Salton. 1989. *Automatic Text Processing.* Addison-Wesley.

Shoham, Y. 1993. Agent-oriented programming. *Artificial Intelligence* 60:51–92.