Yenta: A Multi-Agent, Referral-Based Matchmaking System

Leonard N. Foner

MIT Media Lab
20 Ames St, E15-305
Cambridge, MA 02139
foner@media.mit.edu
617/253-9601

Extended Abstract

Many important and useful applications for software agents require multiple agents on a network that communicate with each other. Such agents must find each other and perform a useful joint computation without having to know about every other such agent on the network. As an example, this extended abstract describes a *matchmaker* system, called *Yenta*, designed to find people with similar interests and introduce them to each other. The matchmaker is designed to introduce *everyone*, unlike conventional Internet media which only allow those who take the time to *speak* in public to be known.

Though most people think of romantic matchmaking when the term "matchmaking" is used, there are many other reasons for doing such matches. For instance, it is often the case that companies or research labs have two or more people working on similar projects, without ever realizing it, simply because no one tells *every* person he or she meets about *everything* he or she is working on. Yenta, which *does* try to do just that, can help make serendipitous matches for researchers and employees who might, by virtue of chance, the layout of their building, or geographic separation, never have thought to ask if they shared some particular interest.

Yenta uses a highly decentralized model, in which there is *no* central server. Such an architecture has important advantages in performance [2], since there is no central site to experience quadratic or worse slowdowns as the number of users goes up, and in security [1][3], since there is no single site whose compromise will threaten all users' privacy.

Instead of a central server, each user runs their own copy of Yenta. Each copy of Yenta then reads those mail messages and files allowed by its user, and builds a profile of the user's various interests by converting the text of the user's various documents into weighted vectors of keywords. These keyword vectors are then used both to cluster individual messages and files into *granules* which describe individual topics in which the user has an interest, and in the inter-Yenta protocol that clumps several users into *clusters*. Users whose Yentas are all in the same cluster share an interest; such users may then be introduced to each other, may send messages into the cluster which most or all of its participants will see, and so forth. Any given user is presumed to have multiple interests simultaneously, and hence will probably be in multiple clusters simultaneously.

In order to form these cluster of users who share interests, Yenta uses a *referral algorithm* so that each Yenta need not contact a large number of others. Instead, each Yenta maintains two caches, its *cluster cache*, which records which other Yentas on the network are already known to be in some cluster with this Yenta, and its *rumor cache*, which is a simple cache of the last few Yentas which have been heard from (typically 50). Whenever the user's Yenta contacts another, it asks the foreign Yenta, in essence, "Given my interests, who in your rumor cache seems the best match for me—even if it's not a great match." The user's Yenta then contacts *that* Yenta and repeats the query, hence using a word-of-mouth referral chain to find better matches.

Simulation results [2] have shown that this algorithm yields acceptable performance in time, cache size, and number of messages exchanged, and does not need any sort of central server to function.

Since Yenta deals with a large amount of potentially sensitive information (e.g., all of the user's mail and files), a large part of its design goes toward making a system secure enough that users can trust it. Part of the solution is to have no central server, which is simultaneously a performance and robustness improvement (avoiding a single point of failure) and proof against a single-site attack or subpoena. The rest of the solution makes extensive use of cryptography [3], which allows secure reputation managment and identification; confidentiality from eavesdroppers; security against spoofing, message modification, and spamming; and assurance, through public peer-review of cryptographically-signed source code, that the distribution itself is not compromised.

See http://foner.www.media.mit.edu/people/foner/Yenta/ for more information.

References

[1] Foner, Leonard, "Clustering and Information Sharing in an Ecology of Cooperating Agents, or How to Gossip without Spilling the Beans," *Proceedings of the Conference on Computers, Freedom, and Privacy '95 Student Paper Winner*, Burlingame, CA, 1995.

[2] Foner, Leonard, "A Multi-Agent Referral System for Matchmaking," *PAAM '96 Proceedings*, London, England, 1996.

[3] Foner, Leonard, "A Security Architecture for a Multi-Agent Matchmaker," *ICMAS '96*, Kyoto, Japan, 1996.