SOAP: Live Recommendations through Social Agents

Hui Guo
GMD-FIT.CSCW Institute for Information Technology
D-53754 Sankt Augustin

Introduction

Less than five years old, the concept of collaborative filtering has already spawned dozens of research prototypes, experimental proprietary systems, and even a few commercially available systems. Our project at the CSCW group of GMD was set up to create an open distributed platform that supports various Web-situated applications through a wealth of interacting software agents. The first application SOAP built upon this platform is designed to provide people with "live bookmarks" by combining content-filtering search engines with techniques from recommender systems [Resnick&Varian'97], so as to match people's interest and to support social construction of knowledge. In the following, we give an outline of our system comparing it with others regarding well-known issues such as incentive and privacy, cold-start, etc. We also briefly introduce the underlying infrastructure and system status.

Agent-based Social Filtering for Recommendation

SOAP is a social agent system [Voss&Kreifelts'97] which consists of various type of agents mediating between different people, groups and the Web. Within this system, users can input queries and get bookmarks back, and are allowed to cluster, annotate and rate these periodically updated bookmarks, as well as form groups to share bookmarks and annotations. The system can match users' interests against each other in terms of different levels of context (query, task, and group), make recommendations accordingly, and make it possible to aggregate knowledge out of user ratings and annotations. The system functionality is basically supported by several types of communicative agents: user agent, task agent, query agent, recommender agent and search agent. These agents have their own knowledge about the user or the environment, and interact with each other through a set of uniform performatives to exchange information in order to achieve their specific tasks.

Within SOAP, each registered user has his/her own unique user agent, which accepts the user's queries, then initiates task agents which in turn distribute task to individual query agents. These query agents request services from a search agent—a service wrapping popular search engines like AltaVista, Infoseek—and a recommender agent which implicitly collects all ratings and annotations from users and performs matching and recommending. After receiving recommendations from these services, query agent, task agent and user agent will cooperatively merge, sort, filter, combine, cluster these information by relevance and present tailored results to users. By using this system, users can iteratively input and refine queries, rate the resulting URLs between -2 (very bad) and 2 (very good), make annotations, review recommendations from others who have similar interests, and gradually obtain a valuable bookmark list with respect to a specific task context which groups all queries, bookmarks, ratings, annotations. This bookmarks list is alive and grows, for the system will autonomously fill in new recommendations based on a push model.

To provide a richer and more constrained context for collaborative filtering, we propose a group agent, which stores queries, results, ratings and annotations of group members like a recommender agent. A group agent allows users to annotate and share bookmarks, matches users' interest and recommends highly-rated bookmarks along with annotations. By cooperating with other information resources like search agents, recommender agents, and other public group agents, a group agent behaves like a space for users to share recommendations and construct personalized views of group bookmark collections. Since users can only join a group by invitation, their identity is visible with regard to ratings or annotations within the group. This prevents non-serious anonymous ratings and ensures better recommendations.

A well-known problem of recommender system is the cold start problem. A recommender system can produce good recommendations only after it has accumulated a large set of ratings. In our system, a bookmark agent is introduced which collects and exploits publicly available bookmark pages by transforming them into the internal recommendation format for use by the recommender agent.

In order to protect the users' privacy appropriately, each information object which may be transmitted to other users as part of a recommendation has privacy-related access control. A typical example is the annotation. A user can make annotations anonymously, which means the user name won't be visible to whoever might get this annotated recommendation. Privacy control is also defined with respect to context, e.g., if a task is made private, all the queries under this task will be private by default as well.

Also, in some recommender systems, there exists an “vote early and often” phenomenon, resulting in recommendations based on faked ratings. Such user actions are detected and inhibited by the user agent in SOAP: all the highly-rated bookmarks are sent to the public recommender agent only once and repetitive ratings of the same bookmark will be simply ignored by the user agent. The user agent could also fetch each bookmark and perform content-based filtering to filter out apparently inserious ratings.

The incentive (or cost/benefit) problem is addressed in SOAP in two ways. First, we keep the rating overhead for users low by interpreting users’ actions as implicit votings: our system allows user to discard a URL or to annotate it without rating, and interprets this as negative or positive feedback, respectively. Second, the SOAP system provides a better quality of information retrieved than public search engines by exploiting human judgements and recommendations, so there is a basic benefit with no cost of rating and annotating. For mutual sharing of costs (and benefits), users are encouraged to rate and annotate, especially in the context of specific user groups.

SOAP Implementation

As an application that allows users to assess Web pages and recommend them to other users and groups, SOAP is implemented on top of an open infrastructure. This infrastructure is a distributed platform for interacting software agents which provides a runtime environment with basic functionality, such as a unique agent naming /addressing schema, a message-based agent communication mechanism, fault recovery, persistency of agents, multiple agent accounting and a distributed directory service for agents. This platform is composed of the kernel layer—agent engine that hosts multiple agents and represents the basic runtime environment with kernel services like agent creation, termination, messaging, etc.—and a service layer that implements system services like naming and alarming service. Both application and infrastructure are implemented in Java.

Status and Future Research

The first prototype released for tests within our research group at GMD in September’97 included agents which realized the user interface, tasks and queries, and agents providing information by contacting search engines or collecting recommendations. This prototype is intended for demonstration, exploratory use, and evaluation in cooperation with an industrial partner from the oil business. Prospective users are members of project teams operating in oil field development. Team members are usually spread around the world, and may belong to several teams at the same time. Information retrieval and exchange is central to their work, and it has to occur both in similar areas or using similar techniques.

For next release, we consider to design agents which provide interface to legacy information system and shared workspace system like BSCW [Bentley et al.’97]. Compared with popular systems like JASPER [Davies et al.’95], PHOAKS [Terveen et al.’97], etc., SOAP is unique in its combination of functionality, and it can be easily extended to offer more, such as recommending users with similar interests. With SOAP, we hope to tackle two important problems, missing context and cold start. Missing context can be supplied by relating URLs to topics and groups. Cold start problems should by avoided by starting with published bookmark collections, by using search engines to introduce new material without any personal overhead. In addition, the scalability of SOAP is expected to be guaranteed by distributing required functionality over individual agents which are distributed over networks. In the context of SOAP, more problems arise such as matching, clustering, use of thesauri to capture group-specific terminology, and creation of summary queries as operational definitions of tasks, and need further investigation in later research.

References

[Bentley et al.’97] Bentley, R., Applet. W., Busbach, U., Hinrichs, E., Kerr, D., Sikkel, K., Trevor, J., Woetzel, G. “Basis support for cooperative work on the World Wide Web,”Int.J. Human Computer Studies 46(‘97), 827-846.

[Davies et al.’95] Davies, J., Weeks, R., Revett, M. “JASPER: Communicating information agents for the WWW,” in *Proc. 4th Int. World Wide Web Conf.* (Boston MA, Dec.1995), World Wide Web Journal Vol. 1, 1, O’Reilly, Sebastopol CA, 1995, pp.473-482.

[Resnick&Varian’97] Resnick, P., Varian, H.R. “Recommender Systems”, *Comm. ACM* 40, 3 (1997), 56-58

[Terveen et al.’97] Terveen, L. Hill, W. Amento, B. McDonald, D. “PHOAKS: A system for sharing Recommendations”, *Comm. ACM* 40, 3(1997), 59-65

[Voss&Kreifelts’97] Voss, A., Kreifelts, Th. “SOAP: Social Agents Providing People with Useful Information”, to appear in *Proc. GROUP97*, Phoenix, Arizona, Nov. 1997.