Modeling User Preferences and Mediating Agents in Electronic Commerce

Mehdi Dastani[1], Nico Jacobs[2], Catholijn M. Jonker[1], and Jan Treur[2]

[1]Department of Artificial Intelligence, Vrije Universiteit Amsterdam,
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
{mehdi,jonker,treur}@cs.vu.nl
http://www.cs.vu.nl/~{mehdi,jonker.treur}

[+] Dept. of Computer Science, Katholieke Universiteit Leuven,
Celestijnenlaan 200A, B-3001 Heverlee, Belgium
Nico.Jacobs@cs.kuleuven.ac.be
http://www.cs.kuleuven.ac.be/~nico

Abstract. *An important ingredient in agent-mediated Electronic Commerce is the presence of intelligent mediating agents that assist Electronic Commerce participants (e.g., individual users, other agents, organisations). These mediating agents are in principle autonomous agents that will interact with their environments (e.g. other agents and web-servers) on behalf of participants who have delegated tasks to them. For mediating agents a (preference) model of participants is indispensable. In this paper, a generic mediating agent architecture is introduced. Furthermore, we discuss our view of user preference modeling and its need in agent-mediated electronic commerce. We survey the state of the art in the field of preference modeling and suggest that the preferences of electronic commerce participants can be modelled by learning from their behaviour. In particular, we employ an existing machine learning method called inductive logic programming (ILP). We argue that this method can be used by mediating agents to detect regularities in the behaviour of the involved participants and induce hypotheses about their preferences automatically. Finally, we discuss some advantages and disadvantages of using inductive logic programming as a method for learning user preferences and compare this method with other approaches.*

1 Introduction

The explosive growth of electronic markets and retail Electronic Commerce has resulted in an overload of online information and products. The effectivity and success of this market depends on the amount of automated Electronic Commerce processes and services that are available online. Finding, comparing, buying, selling and customising items via the World Wide Web, automatic negotiation and personalised recommendation services are examples of such processes and services. Some of these processes and services are already available on the World Wide Web though in limited forms. For example, search engines like Altavista and Yahoo help people to locate items on the web and online shop sites such as Jango [44], Amazon [39] and Moviefinder [46] offer personal recommendation services to advise their customers about products that may be interesting to them. Also, online auction sites such as eBay

F. Dignum and C. Sierra (Eds.): Agent Mediated Elec. Commerce, LNAI 1991, pp. 163-193, 2001.
© Springer-Verlag Berlin Heidelberg 2001

[42] and AuctionBot [40] provide automatic bid proposal services such that a customer needs not to be online during a chosen auction. Finally, in the forthcoming marketplaces such as MarketMaker [45] users can create agents and delegate various tasks such as buying, selling, and searching items to them. These agents are able to negotiate with each other in order to perform delegated tasks. These services help customers to avoid the large search space of available items or the need to be involved in all required activities.

In general, to support users on the World Wide Web, various types of agents can be developed. For example, to support brokering processes in electronic commerce, agents can be developed that support a user offering products (or services) at the World Wide Web. Also, agents can be developed that support a user searching for information or products within the scope of user's interest. Of course, agents can be developed to combine both functionalities as well. Moreover, mediating agents can be developed that communicate with both agents, i.e. with agents that provide information or products and with agents that ask for information or products. Recently a few applications of mediating agents have been addressed for this area; for example, see [9], [10], [22], [27], [33], [35]. In general, applications like these are implemented in an ad hoc fashion without an explicit design at a conceptual level.

The aim of this paper is twofold. On the one hand, a generic agent architecture for mediating agents acting in brokering processes is introduced which has been designed in a principled manner, using the compositional development method for multi-agent systems DESIRE. The agent architecture can be instantiated by adding specific types of knowledge to support functionalities and behaviour required. Depending on the choice of these requirements, an agent is created for a specific application by including the appropriate types of knowledge. For example, a search agent with functionality restricted to (incidental) search for information upon a user's request can be built by adding only knowledge needed for this task. Such an agent, for example, is not able to store and maintain the user's query or information that has been found, nor is it able to provide information to other agents. If these functionalities are required as well, the necessary types of knowledge have to be added. On the other hand, we present an overview of some existing approaches in preference *modeling* and briefly discuss them. It is discussed that each approach is appropriate for certain classes of applications. Finally, we explore in more detail the use of Inductive Logic Programming (ILP) as a possible method for automatic preference *modeling*. We explain how this method can be employed in order to induce preference models. The (dis)advantages of this method are discussed and some experimental results are presented.

In Section 2 an example problem domain for brokering processes is sketched. Section 3 introduces the design of the generic architecture for mediating agents. The different types of knowledge are presented in Section 4. In Section 5 the behaviour of the system is analysed by giving an overview of which types of knowledge are needed for which types of basic functionalities. In Section 6 an overview is given of recent literature on preference *modeling*, showing the need for an automatic approach to user preference *modeling*. Finally, Section 7 shows that Inductive Logic Programming is a possible technique for constructing preference models and may be useful as an algorithm for the production of a classification tree that can be used to match products against preferences.

2 Electronic Commerce and Brokering

The process of brokering as often occurs in Electronic Commerce involves a number of agents. A provider agent that provides information about products to other (human or computer) agents may support a user offering products. A user looking for products may be supported by a personal assistant agent that takes its user's queries and contacts other agents or looks at the Web directly to find information on products within the user's scope of interest. Such a personal assistant agent may contact either provider agents immediately, or mediating agents, which in turn have contact with provider agents, or other mediating agents. Depending on the application, the chain of agents involved may include zero or more mediating agents.

The domain analysed for the agent architecture presented here is the domain of brokering (scientific) papers. Although this domain might not be considered as a real electronic commerce application (for instance because electronic commerce applications usually involve money as an important attribute of domain items), we have chosen it because of the easy access to data needed to do experiments. Moreover, we believe that our approach is general and thus can be applied to any domain consisting of any set of attributes. In particular, we will show that both proposed learning method and the agent architecture are flexible enough to be applied to real electronic commerce applications.

The domain of scientific papers has a number of aspects in common with other domains within the area of electronic commerce. The task of a provider agent is to inform other researchers on papers available on Internet (a marketing aspect). For example, an agent related to a Web site of a research group announces new papers included in their Web site. If a researcher is looking for a paper with certain characteristics (scope), a personal assistant agent can ask other agents for information on papers with these characteristics. To be able to tune the information provided to users, a number of scopes of interest can be maintained for each of the users. For example, one of the users may be interested in papers on certain topics, such as work flow management systems, but also in papers on agents and the World Wide Web.

Topics can be basic (e.g., 'work flow management systems', or 'agents', or 'World Wide Web'), or a combination of a number of topics (e.g., 'agents and World Wide Web'). In the latter case the user interest is limited to papers which address both topics. Moreover, if it is added that the user is only interested in papers from the years 1995 to 1997, then either year in the range 1995-1997 is meant. Topics can be matched with, for instance, the set of keywords of a paper, or with the abstract, or the paper as a whole. In some disciplines, such as Medicine, an ontology of topics has been developed that serves more or less as a standard. Besides topics also other attributes of papers can be used to define a scope of interest, for example an author, a year, a research group, et cetera. These attributes can also be used in combination with each other. For this example, a shared ontology of topics is assumed. All agents in the brokering process express their information and interests using this shared ontology. It is assumed that the following attributes of a paper are available and can be used: title, authors, affiliation(s) of the authors, location on the World Wide Web where it can be found, topics covered by the paper, abstract, year, and reference. This information can be used to identify papers that are of interest to a user, but also forms the source for the information that can be provided to a user when a paper is proposed to him or her.

3 Design of the Generic Mediating Agent

The generic mediating agent presented in this paper offers a reusable agent that can be applied (reused) in the context of a multi-agent system which can take different forms. One simple possibility is that the mediating agent serves as a personal assistant representing a buyer and communicates with this user and with other software agents that represent sellers. The generic mediating agent supports the user profiling involved, but also maintenance of information on sellers. More complex possibilities can involve, in addition, communication between buyer personal assistant agents, for example, to combine requests and form coalitions. The generic mediating agent does not exclude this possibility, but no explicit structures have been added yet to support this. Another possibility is to use the generic mediating agent as the basis for a broker agent which communicates both with buyer representative agents and seller representative agents.

For the design of the generic mediating agent the following main aspects are considered: process composition, knowledge composition, and relations between knowledge and process composition, as discussed in [6]. A compositional generic agent model (introduced in [8]), supporting the weak agency notion (cf. [37]) is used. At the highest abstraction level within an agent, a number of processes can be distinguished that support interaction with the other agents. First, a process that manages communication with other agents, modelled by the component agent interaction management in *Figure* 1. This component analyses incoming information and determines which other processes within the agent need the communicated information. Moreover, outgoing communication is prepared. Next, the agent needs to maintain information (including indications of specific interests and preference models built over time) on the other agents with whom it co-operates: maintenance of agent information. The component maintenance of world information is included to store world information (e.g., information on attributes of products). The process own process control defines different characteristics of the agent and determines foci for behaviour. The component world interaction management is included to model interaction with the world (with the World Wide Web world, in the example application domain): initiating observations and receiving observation results.

The agent processes discussed above are generic agent processes. Many agents perform these processes. In addition, often agent-specific processes are needed: to perform tasks specific to one agent, for example directly related to a specific domain of application. In the current example the agent has to determine proposals for other agents. In this process information on available products (communicated by information providing agents and kept in the component maintenance of world information), and about the interests of agents and their preference models (kept in the component maintenance of agent information), is combined to determine which agents might be interested in which products. For the mediating agent this agent-specific task is called determine proposals. *Figure* 1 depicts how the mediating agent is composed of its components.

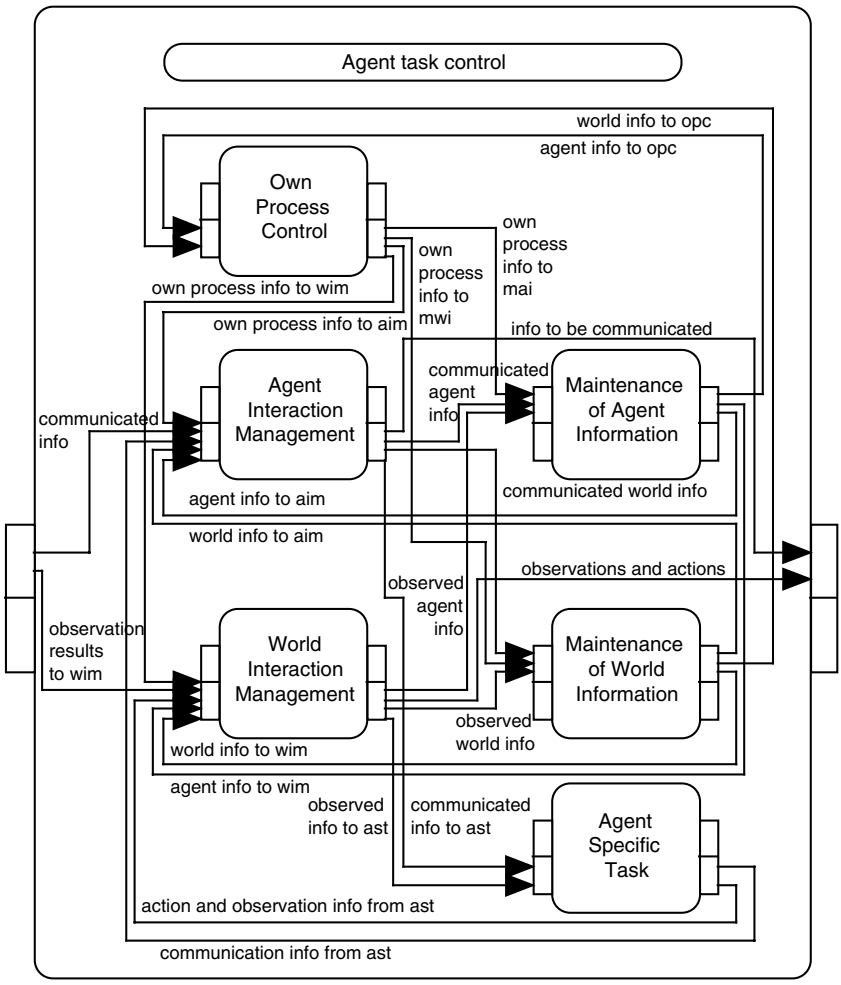

Fig. 1. Composition at the highest level within the mediating agent

Part of the exchange of information within the generic agent model can be described as follows. The mediating agent needs input about interests, put forward by agents, and information about attributes and evaluations of available products that are communicated by information providing agents. It produces output for information agents about proposed products and the attributes of these products. Moreover, it produces output for information provider agents about interests. In the information structures (called information types) that express communication information, the subject of the communication and the agent from or to whom the communication is directed are expressed. This means that communication information consists of statements about the subjects that are communicated.

Within the mediating agent, the component own process control uses belief input information and generates focus information: to focus on a scope of interest to be

given a preferential treatment, i.e., pro-active behaviour will be shown with respect to this focus. The component agent interaction management has the same input information as the agent (incoming communication), extended with belief info and focus info. The output generated includes part of the output for the agent as a whole (outgoing communication), extended with maintenance info (information on the world and other agents that is to be stored within the agent), which is used to prepare the storage of communicated world and agent information. Information on attributes of products is stored in the component maintenance of world information. In the same manner, the beliefs of the agent with respect to other agents' profiles (provider attribute info, preference model info, and interests) are stored in maintenance of agent information. The agent specific task determine proposal uses information on product attributes, preference models, and agent interests as input to generate proposals as output. For reasons of space limitation the generic and domain-specific information types within the agent model are not presented; for more details [19].

4 Generic and Domain Specific Knowledge

The different knowledge abstraction levels introduced for information types can also be exploited to structure the knowledge. Abstract knowledge can be formulated in terms of scopes, abstracting from attributes and values. Other more specific knowledge is used to perform the abstraction step: it can be used to derive conclusions in terms of scopes from input in terms of attributes and values. The knowledge bases are discussed below in the context of the component in which they are used. Knowledge bases not specified in this paper can be found in [19].

4.1 Agent Specific Task: Determine Proposals

To determine proposals fitting a given scope of interest, information on products has to be compared to this scope of interest. To this end, the information on products, expressed in terms of their attributes has to be aggregated to information in terms of scopes. This can be derived using two knowledge bases, attribute and scope kb, which defines the relations between attributes and scopes in general, and product scope abstraction kb, which identifies for which scope(s) a product is relevant. The composition of the knowledge in these two knowledge bases supports reuse. For example, if in one of the two knowledge bases, modifications are made, the other knowledge base still can be used. Moreover, the first knowledge base is specified independent of knowledge about products. It can be (re)used within the component maintenance of agent information as well, as will be shown below. Given information on the scopes of products, by the knowledge base strict match kb it is defined how proposals to agents can be generated by matching the scopes of products and the scopes in which an agent is interested. For strict matching it consists of only one element.

Knowledge base strict match kb
if interested_in(A:AGENT, S:SCOPE)
and in_scope(P:PRODUCT, S:SCOPE)
then is_possibly_interesting_for(P:PRODUCT, A:AGENT, S:SCOPE)

This knowledge simply states that if a product is in a scope an agent is interested in, then this product is possibly interesting for this agent. Alternative knowledge bases can be used for non-strict matching. Using a method to determine a predicted rating (e.g., the classification tree that is constructed by the learning algorithm described in Section 7), the products that are possibly interesting for a user are classified with a predicted rating. In a selection phase only those products that have a sufficiently high predicted rating are presented to the user.

Knowledge base proposal_selection kb
if is_possibly_interesting_for(P:PRODUCT, A:AGENT, S:SCOPE)
and has_acceptable_predicted_rating(P:PRODUCT, A:AGENT)
then is_interesting_for(P:PRODUCT, A:AGENT, S:SCOPE)

By adapting the predicted rating to his or her own preferences the user influences the learning method contained within the component maintenance of agent information and thereby indirectly influences the predicted rating knowledge (e.g., the form of the classification tree discussed in Section 7) used within the current component (determine proposals).

4.2 Agent Interaction Management

The component agent interaction management makes use of five knowledge bases: (1) for incoming communication from agents asking for information on products, (2) incoming communication from agents giving their evaluation of products (necessary for constructing preference models), (3) incoming communication from agents providing information, (4) outgoing communication to agents interested in information on products, and (5) outgoing communication to agents providing information.

4.2.1 Incoming Communication

If an agent communicates her or his interests to a mediating agent, then this information is identified as new agent interest information that is currently believed (which can be forgotten after the agent has reacted on it: knowledge base agent interest identification kb) or that has to be stored (in which case it can be remembered later: knowledge base agent interest maintenance identification kb). A condition for storage of interests information is that the type of contract is persistent. For agents with a weaker type of contract no requests are stored, and instead of building a user specific preference model a default preference model can be used.

Knowledge base agent interest maintenance identification kb

if communicated_by(interest(S:SCOPE), V:SIGN, A:AGENT)
and belief(has_contract(A:AGENT, contract_type(persistent, Y)), pos)
then new_agent_info(interested_in(A:AGENT, S:SCOPE), V:SIGN)

If an agent communicates that he or she wants to subscribe for a contract of a certain type, then this information is identified as new contract information that has to be stored. This identification makes use of the following knowledge base.

Knowledge base subscription identification kb
if communicated_by(subscription_for(C:CONTRACT_TYPE),
V:SIGN, A:AGENT)
then new_agent_info(has_contract(A:AGENT, C:CONTRACT_TYPE), V:SIGN)

If an agent has a persistent type of contract with the mediating agent, then evaluations of products given by that agent have to be identified, so that they can be used in the maintenance of his/her preference model.

Knowledge base agent preference information identification kb

if communicated_by(is_rated_as(P: PRODUCT, R: RATING), V:SIGN, A:AGENT)
and belief(has_contract(A:AGENT, contract_type(persistent, Y)), pos)
then new_agent_info(is_rated_as_by(P: PRODUCT, R: RATING, A:AGENT), V:SIGN)

If an agent communicates information about products it provides, this incoming information is analysed, new world information is identified as new information that can be used immediately and forgotten afterwards (knowledge base provider info identification kb), or has to be stored (knowledge base provider info maintenance identification kb). If an agent communicates information about products it provides, this incoming information can also be analysed, in order to obtain new agent information on the scopes of the information the agent (apparently) can provide. This is expressed by:

Knowledge base provider scope maintenance identification kb:

if communicated_by(attribute_has_value(P:PRODUCT, A:ATTRIBUTE, V:VALUE), pos, A:INFO_PROVIDER)
then new_agent_info(can_provide(A:INFO_PROVIDER, A:ATTRIBUTE, V:VALUE), pos)

4.2.2 Outgoing Communication

New information (product identification, scope, predicted rating, or attribute information) on a product that may be interesting for an agent is communicated to this agent. This is expressed in the following knowledge base:

Knowledge base proposal communication kb:

if belief (is_interesting_for(P:PRODUCT, A:AGENT, S:SCOPE), pos)
and belief(attribute_has_value(P:PRODUCT, A:ATTRIBUTE, V:VALUE), pos)
and belief(product_has_predicted_rating_for(P:PRODUCT, R: RATING, A: AGENT), pos)
then to_be_communicated _to(is_interesting(P:PRODUCT, S:SCOPE), pos, A:AGENT)
and to_be_communicated_to(attribute_has_value(P:PRODUCT, A:ATTRIBUTE, V:VALUE), pos, A:AGENT)
and to_be_communicated_to(product_has_predicted_rating(P:PRODUCT, R: RATING), pos, A: AGENT)

The agent only communicates to an information provider if a scope has been taken as a focus, and if the information provider can provide products on this scope. This is expressed by:

Knowledge base info provider request kb

if in_search_focus(S:SCOPE)
and belief(can_provide_scope(A:AGENT, S:SCOPE), pos)
then to_be_communicated _to(interest(S:SCOPE), pos, A:AGENT)

4.3 Own Process Control

The types of proposals to be determined and the scopes on which to focus search are determined by means of the knowledge base focus kb, as indicated by the following knowledge base. The knowledge base focus kb is used within own process control component. For example, in the first knowledge element it is expressed that for an agent with a contract of any type, proposals will be determined that fit the agent's interests. This is in contrast with, for example, the second knowledge element which expresses that only for agents with a persistent contract type, their scopes of interests will be chosen as persisting search foci (otherwise these scopes of interest will be forgotten after being handled).

Knowledge base focus kb

if belief(has_contract(A:AGENT, C:CONTRACT:TYPE))
and belief(interested_in (A:AGENT, S:SCOPE), pos)
then product_to_be determined(in_scope (P:PRODUCT, S:SCOPE))
and proposal_to_be determined(is_interesting_for(P:PRODUCT, A:AGENT, S:SCOPE))

if belief(has_contract(A:AGENT, contract_type(persisting, search_for_info)))
and belief(interested_in (A:AGENT, S:SCOPE), pos)
then in_persisting_search_focus(S:SCOPE)

if belief(has_contract(A:AGENT, contract_type(incidental, search_for_info)))
and belief(interested_in (A:AGENT, S:SCOPE), pos)
and not search_focus_processed_for(S:SCOPE, A:AGENT)
then in_incidental_search_focus(S:SCOPE)
and search_focus_chosen_for(S:SCOPE, A:AGENT)

if in_persistent_search_focus(S:SCOPE)
then in_ search_focus(S:SCOPE)

if in_incidental_search_focus(S:SCOPE)
then in_ search_focus(S:SCOPE)

if in_ search_focus(S:SCOPE)
then provider_to_be_determined_for(S:SCOPE)

4.4 World Interaction Management

The component world interaction management allows the agent to look for information by observation. This entails generation of observations to be performed and obtaining the observation results. The obtained observation results can be used incidentally after which the information is forgotten (using knowledge base observation info identification kb) or maintained to be used later as well (using knowledge base observation maintenance identification kb), similar to agent

interaction management. The agent only observes if a scope has been taken as a focus. This is expressed using knowledge base observation initiative kb.

Knowledge base observation initiative kb
if in_search_focus(S:SCOPE)
then to_be_observed(S:SCOPE)

The actual execution of the observation does not take place within the agent, but in the external world. As part of the external world an engine can be used to search for products matching the pattern defined by the specified scope. The result of such an observation will be all information of any product that matches the scope. The knowledge base specified above is kept rather simple. To avoid too frequent repetition of observation, more sophisticated knowledge can be specified.

4.5 Maintenance of World and Maintenance of Agent Information

In principle, the components maintenance of world information and maintenance of agent information store information. The knowledge base attribute and scope kb defined above is also used in the component maintenance of agent information. In addition, the knowledge base provider scope abstraction kb is defined; it is similar to the knowledge base product scope abstraction kb mentioned above. But most importantly with respect to user preference *modeling*, the product evaluations (as given by the user in response to products presented to him/her) are used to adapt the user preference model within the component maintenance of agent information. An example of a technique to be used is described in Section 7.

5 The Behaviour

The behaviour of the mediating agent can be analysed in different ways. One way is to consider its basic functionalities with respect to its brokering task, and use these as building blocks to obtain behaviour. For example, its behaviour in terms of the weak notions of agency (autonomy, social ability, reactivity, and pro-activity) can be determined in terms of basic functionalities. Moreover, basic functionalities can be related to knowledge bases that are available within the agent. Using these two relationships, a relation can be identified between behaviour and available knowledge within the agent.

5.1 Basic Functionalities Depending on the Agent's Knowledge

The mediating agent shows behaviour depending on certain basic functionalities. For the agent model presented, these basic functionalities have been specified in a declarative manner by the agent's knowledge. For each of the basic properties of the agent it has been established which knowledge bases are required. By varying the choice of knowledge for the agent, different types of agents can be designed.

1. *Observation of information available within a certain part of the world*

Observation requires the ability to initiate observations, specified in the knowledge base observation initiative kb, and the ability to identify the information resulting from an observation, specified in the knowledge base observation info identification kb. Both knowledge bases can be used within the component world interaction management.

2. *Communication with agents asking for information on products*

The basic functionality to communicate with agents asking for information on products requires the processing of incoming communication of asking agents and preparation of outgoing information. The incoming information may refer to scopes of interests of the asking agent, evaluations of products, or to subscription. The communicated scopes of interest are identified using the knowledge base agent interest identification kb. That an agent is providing feedback information regarding products is identified using knowledge base agent preference information identification kb. Incoming communication on subscription is identified using knowledge base subscription identification kb. Outgoing communication containing product information to agents that ask for information is prepared using knowledge base proposal communication info. All these knowledge bases are used within the component agent interaction management.

3. *Communication with agents providing information on products*

Communicated information on products can be processed in two different ways. First, the product information can be identified, using knowledge base provider info identification kb. Second, from the fact that information is provided on a product with certain characteristics, it can be abstracted (from the given product) that this provider is able to offer (at least some) products with these characteristics in general. This is done using knowledge base provider scope identification kb. Communication to an agent that may be able to provide information is prepared using knowledge base provider request kb. All these knowledge bases are used within component agent interaction management.

4. *Maintenance of acquired information on products*

The agent can identify that all communicated information on products has to be stored, using knowledge base provider info maintenance identification kb within component agent interaction management. Moreover, by knowledge base observation info maintenance identification kb, within component world interaction management, new observation results on products to be stored can be identified.

5. *Maintenance of scopes of interest of agents and the preference models*

The agent can identify that the incoming requests of agents are to be maintained. This functionality is specified by the knowledge base agent interest maintenance identification, used within component agent interaction management. Feedback information regarding products is identified using knowledge base agent preference information identification kb, this knowledge base is also used within component agent interaction management. The feedback information itself is used for the maintenance of user preference models; for an example see Section 7.

6. *Maintenance of scopes of products agents can provide*

Scopes of information agents can provide are stored, if the incoming communication is handled in an appropriate way using knowledge base provider scope maintenance identification kb, used within component agent interaction management.

7. *Own control*

Control of the agent's own processes is defined by the knowledge base focus kb, used within component own process control.

8. *Determining matches between products and scopes of interests*

To determine matches between products and scopes of interest the agent can use the knowledge bases attribute and scope kb, product scope abstraction kb, proposal selection kb, and strict match kb within component determine proposals.

basic functionality	*knowledge specifying functionality*	*in component*
1. observation	observation initiative kb observation info identification kb	WIM WIM
2. communication with agents asking for information	agent interest identification kb subscription identification kb proposal communication kb agent preference information identification kb	AIM AIM AIM AIM
3. communication with agents providing information	provider info identification kb provider scope identification kb provider request kb	AIM AIM AIM
4. maintenance of product information	observation info maintenance identification kb provider info maintenance identification kb	WIM AIM AIM
5. maintenance of scopes of interest and preference models	agent interest maintenance identification kb agent preference information identification kb	AIM AIM
6. maintenance of scopes of products agents can provide	provider scope maintenance identification kb provider scope abstraction kb attribute and scope kb	AIM MAI MAI
7. own control	Focus kb	OPC
8. match between products and scopes of interests	attribute and scope kb product scope abstraction kb strict match kb proposal selection kb	DP DP DP DP

Fig. 2. Relation between basic functionalities and knowledge required

Combinations of these functionalities define specific types of agents. For example, if a provider agent is designed, functionalities 2, 4, 5, 8 may be desired, whereas functionalities 1, 3, 6, 7 could be left out of consideration. If an agent is designed to support a user in finding information on products within a certain scope, functionalities 1, 3, 6, 8 (and perhaps 4) may be desired, whereas 2 and 5 may be less relevant. For a mediating agent, or for an agent that has to play different roles, almost

all functionalities (i.e., 1 to 8) may be desired. The generic agent architecture introduced in Sections 3 and 4 can be instantiated in different manners to obtain, among others, the types of agents mentioned. The relation between the agent's basic functionalities, its knowledge, and where the knowledge is used is summarised in the table in *Figure* 2.

5.2 Reactive, Pro-active, and Other Forms of Behaviour

Depending on the choices made, the mediating agent can show reactive behaviour towards agents asking for information on products and provider agents.

In reaction to an agent that asks for products within a certain scope, the mediating agent determines which of the products it knows, fit to this scope, using either an already known preference model for that agent, or a default preference model. The available information on the resulting products is communicated to the agent (e.g., author, title, year, topics, abstract, location, and reference).

Once an agent interest is known to the mediating agent, it is reactive with respect to any information providing agent that announces a product that fits the agent's scope, and has a sufficiently high predicted rating. In such a case the information on this product is communicated to this agent (i.e., to all relevant agents).

Pro-active behaviour occurs when the mediating agent has as a characteristic that it is pro-active with respect to certain agents. A pro-active mediating agent, from time to time, takes the initiative to ask provider agents for information on products which match some of its subscribed request profiles. It may focus on an agent's scopes of interest and actively select information providing agents and ask them whether they have products that fit in one of these scopes.

The behaviour of the mediating agent may depend on other characteristics of the mediating agent as well. In the above example, the knowledge used within own process control was kept rather simple. It is not difficult to extend this knowledge in such a way that more complex forms of pro-active social behaviour are initiated and controlled. For example, it is also possible that the mediating agent pro-actively determines an expected scope of interest of an agent and proposes products that fit this expected scope of interest.

6 Preference Modeling

In this section, we survey the field of user preference *modeling* and discuss some existing approaches and related working systems. Basically, the preference model of a user can be used to determine how interesting is an item to that user. The preference model of a user can thus be used to select and prioritise items that may be interesting to that user. For example, a user may like French or German cars and prefer to have a German car above a French car. The structure and properties of preference models depend on the application area in which they are used. For example, in multi-attribute decision systems (see [2],[21],[29],[36]) the user preference (utility) for an item is determined in terms of values of various attributes of the item and the preferences of the user towards those attributes (i.e., the importance of those attributes). In other application areas such as recommendation systems, the preference model may be defined either in terms of statistical correlation between users and their rated items or in terms of a set of attribute values that describe the items.

In general, the preferences of a user towards a set of items can be defined in terms of information concerning either the content of the items (content information) or the use of the items by a society of users (collaborative or social information). Roughly speaking, in the content-based approach a user is thought to like an item if the item is similar to other items that are liked by the user while in the collaborative-based approach a user is thought to like an item if the user is similar to other users who like the item. In the following, we use the term content-based preference model to indicate user preference models that are defined in terms of the content of items, and the term collaborative-based preference model to refer to user preference models that are defined in terms of collaborative information. In this paper, we assume that both the content information as well as the collaborative information can be used to construct user preference models for various kinds of applications, included automated negotiation in multi-agent systems where a model of user preferences is indispensable.

The construction of a preference model is usually a time consuming and cumbersome job. In applications such as information retrieval, information filtering, or automated integrative negotiation, the user has to express her preferences towards various (combinations of) attributes and attribute values. In other applications such as recommendation systems, a user may be asked to rate several, sometimes hundreds, of items before an item can be recommended. There are various methods to acquire information concerning user preferences. For example, in some systems a user may be asked to fill-out a form consisting of questions (usually a large number of questions) about her preferences every time she uses the system. Instead of forms, systems may also ask a user to answer consecutive multiple-choice questions in an interview-like interaction. Yet, other systems (see [14]) derive the preferences of a user by suggesting an item to the user and ask her to correct this suggestion. The user corrects system's suggestion by indicating why the suggested item does not match her needs. Based on these corrections, preference models of users are constructed or updated. Finally, some systems employ methods to induce the preferences of a user by observing the behaviour of that user over time (see [17],[23],[26],[30]). These methods are usually not intended to fully model user preferences, but to model the more frequent and predictable user preferences. It should be noted that applications that require huge efforts from their users risk to become ineffective and useless (see [24],[25]). Therefore, to model user preferences in an application a balance is to be found between the amount of interaction with the user and the necessary effectiveness of the constructed user profile.

Modeling user preferences on the basis of content or collaborative information can be considered as a learning problem where the aim is to learn the so-called preference function for a certain user. The preference function for a user maps items from a certain domain to some values that express the importance of those items for that user. In this way, the structure of the chosen range is imposed on domain items. It is important to note that various types of preference functions may exist. The type of a preference function characterises the structure of preference model (see [21],[36]). For example, the range of one preference function may be the set of real numbers where the order of real numbers reflects the degree of user interest. The range of another preference function may be the set consisting of two elements: LIKE and DISLIKE. In the first case, a partial order structure is imposed on the items and in the second case a nominal structure is imposed on the items, i.e. the preference function is a classification function.

6.1 Collaborative-Based Preference Modeling

In the collaborative approach the preference model of a user is constructed on the basis of the items that are previously used and rated by that user and the preferences of other users represented as sets of rated items. Intuitively, in the collaborative approach an item is thought to be interesting for a user if other users who have similar taste are interested in that item too. The taste similarity of users is determined by a statistical correlation between users and their rated items. In this way, a group of users that rate items similarly are considered as having similar taste or interest. This approach to user preference *modeling* is often called "word of mouth" propagation. For example, consider the following data table representing the rating (a number between 1 and 10) that four users $U_1,\ldots, U_4$ have assigned to three papers $P_1,\ldots, P_3$. An empty cell in the table indicates that the rating of a person for an item is unknown.

	U_1	U_2	U_3	U_4
P_1	4	-	5	9
P_2	1	9	-	8
P_3	8	1	7	2

It should be clear that persons U_1 and U_3 have similar rating and therefore can be considered as having similar taste. Likewise, U_2 and U_4 show similar taste. These similarities can be used to predict the interest of users towards papers for which the rating is unknown. For example, the taste similarity between U_1 and U_3 suggests that the interest of U_3 towards the second paper is low, while the taste similarity between U_2 and U_4 may suggest that the interest of U_2 towards the first paper is high.

Collaborative-based preference models have been used in retrieval and filtering systems to, respectively, retrieve and filter available items for certain users. In fact, the purpose of these systems is to assist a user by selecting, prioritising, and delivering available items according to the preferences of that user. In this way, the selected items are presented to a user in the order of their relevance for that user. These retrieval and filtering systems are often used as recommendation systems where users are informed about items that may be interesting to them. There have been several collaborative-based recommendation systems introduced in which the preferences of users are modelled automatically. Examples of online recommendation systems that employ collaborative approach are MovieFinder [46] and FireFly [43]. The preferences of a user are modelled automatically by observing the behaviour of that user and applying different statistical methods to the observed behaviour (see [3],[17],[18],[34]).

In collaborative-based recommendation systems, an item can be recommended to a user if the user has already rated a subset of items and thereby has expressed some of her preferences. For this reason, these recommendation systems construct an initial preference model for a new user by asking the user to rate a pre-selected set of items. However, a serious disadvantage of these systems is that new items cannot be

recommended to anyone since items are recommended to users only when they are rated by some users. Note also that the quality of recommendations by systems that are based on collaborative approach increases as the number of users and the number of rated items increase.

6.2 Content-Based Preference Modeling

The content-based approach provides the preference model of a user on the basis of properties and attribute values of the items. Using content-based approach, an item is thought to be interesting for a user if that item has properties or attribute values as predicted by the preference model. It is important to note that, in contrast to the collaborative approach, the content-based approach can be applied only when items can be described in terms of properties and attribute values. Like collaborative-based preference models, the content-based preference models have been used in online recommendation systems such as BargainFinder [41] and Jango [44]. However, unlike collaborative-based preference models, the content-based preference models are also used in applications such as integrative negotiation where the utility function is defined in terms of user preferences towards various attribute values (see [2],[16],[21],[26],[36]).

In general, content-based preference models are constituted by a set of attributes. For example, the set M = {Topic, Author, Year} of paper attributes may constitute the preference model of users for scientific papers. Given a set of attributes constituting the user preference model, the preferences of a user are often modelled by providing some values and rates for each attribute. First, for each attribute a rate may be given to indicate how important is an attribute. Then, for each attribute a set of possible attribute values should be given. Moreover, a rate should be assigned to each possible attribute value to indicate how preferred is that value. The assignment of rates to attribute values depends on the type of attribute values (e.g. nominal, ordinal, interval, and ratio). In fact, for nominal and ordinal attribute values a rate is assigned to each attribute value while for interval and ratio attribute values the inherent order of those values can be used to assign a rate to only a subset of those attribute values. The rate for other attribute values can then be derived by means of the inherent order of attribute values and the assigned rates to the subset of attribute values. For example, given the above set M of paper attributes, the preferences of a user may be modelled by the following set:

{Topic:9 = <Agent:9 , Negotiation:6 , AI:4>,
Author:7 = <Jennings:8 , Zlotkin:8 , Maes:6>,
Year:4 = <1999:9 , 1984:4> }

The numbers attached to attribute names and attribute values indicate user's rates for those attributes names and attribute values, respectively. In this example, the values of the first two attributes (i.e. Topic and Author) have nominal type whereas the values of the third attribute (i.e. Year) have an interval type. Therefore, based on the rate of the two year values (i.e. 1999:9 and 1984:4) and given the internal interval order of year values the rate of other year values can be derived (e.g. 1987:5, 1990:6, 1993:7, etc.).

As the values of attributes may not always be known to a user, the user may also be asked to represent compensation values for the unknown attribute values, i.e. the loss

of an attribute value is compensated by gain in the value of another attribute. The following table is an example of compensation values between paper attributes.

	Topic	Author	Year
Topic	-	0.9	0.3
Author	0.8	-	0.4
Year	0.5	0.5	-

Basically, this table shows how the loss of an attribute value should be gained in terms of other attribute. In this way, the compensation values can be considered as representing user preferences towards interdependencies between various attributes and can be used to determine the user preference towards an item for which some attribute values are missing. Note that these scenarios are quite usual in automated integrative negotiation system (see [2]).

Although content-based preference models play an essential role in both recommendation systems as well as automated negotiation and decision theory, there is little attention in these studies for *modeling*, especially automated *modeling*, of content-based user preferences. An obvious and non-automatic way to model user preferences is the so-called "deep interview" approach. In this approach, the user is asked to answer consecutive multiple-choice questions by means of which item attributes and their values are rated.

A more interesting and semi-automatic way to model user preferences is the so-called "suggestion-correction" approach. This approach assumes a partial preference model of user which may be a default model in the worse case, i.e. when user is new to the system such that nothing is known about user except some default knowledge. For example, in applications such as computer selling systems or a travel agency systems some default knowledge about user such as "need-a-computer" or "want-to-go-to-holiday" can be assumed. Based on partial user preference model the system may suggest an item to that user and, if needed, the user corrects this suggestion by indicating why the suggested item does not satisfy her needs. In the case that the suggested item satisfies the needs of user the system stops. Otherwise, based on user's correction response the system updates the preference model and suggests a new item, etc. This approach is employed in Eugene et. al. (see [14]) where user's correction responses are considered as constraints. The constraints are then linked together to form a network of constraints. The resulting network of constraints represents the preference model of user. Consequently, an item is thought to be interesting for a user when it satisfies the network of constraints that represents the preference model of that user.

Finally, a fully automatic content-based approach to model user preferences is by discovering regularities among properties and attribute values of the used and rated items. Neural networks, genetic algorithms, principle component analysis, and all kinds of inductive learning methods are alternative techniques in automatic content-based preference *modeling*. Although the systems that employ automatic methods

usually expect little effort from users, a new user is expected to make some effort and provide feedback to the system in order to ensure reasonable performance from the start. This is also true for the systems that are based on collaborative-based user preference models. It should also be noted that automatic methods to model user preferences is not only interesting for minimising the effort of users, but it may also be interesting for discovering the preferences of other involved participants. This is especially important for intelligent mediating agents that have to discover the preference models of other involved agents automatically (see Sections 3,4, and 5). Also, in cooperative multi-agent negotiation processes where an agent, who does not have a direct access to the preferences of her negotiating agent, likes to propose a bid that may be interesting to the negotiating agent as well.

In the content-based approach preferences can be modelled independent of preference models of other involved participants. In fact, the preference model of a user can be constructed on the basis of the descriptions of the items for which the user preference are known. Moreover, the quality of the content-based preference models that are provided by automated methods depends on the number of items for which the user preferences are known. This quality is, however, independent of the number of other users or their preferences, as it is the case in collaborative approach. Another characteristic of the content-based approach, in contrast to the collaborative approach, is that a new item can immediately be decided to be interesting for a user without the need of being rated by other users.

6.3 An Integrated Approach

The collaborative-based and content-based approaches do not exclude each other and in fact they can be combined into an integrated approach to model user preferences (see [1]). Such a user preference model will be called integrated user preference model. An integrated user preference model is thus defined in terms of both collaborative as well as content information. In particular, an integrated user preference model is constructed in terms of a predefined set of attributes, as it is the case with the content-based user preference models. However, unlike the content-based preference models, there are two attributes in the integrated approach which are defined in terms of collaborative information. These attributes are called collaborative attributes. One collaborative attribute characterises a user and the second collaborative attribute characterises an item.

In order to construct automatically the integrated preference model of a user, an inductive learning method is applied to a set of data entries (see [1]). Each data entry is an n-tuple of attribute values and represents the information about one user and one item that is liked by that user. Note that one may also consider the set of data entries in which an entry represents the information about one user and one item that is disliked by that user. The value of the collaborative attribute that characterises a user is a set of items that is liked/disliked by that user and the value of the collaborative attribute that characterises an item is a set of users that like/dislike that item. Note that the values of the collaborative attributes are set values: their values are sets instead of individuals. In contrast to the collaborative attribute values, the values of other (non-collaborative) attributes are individual values. Given n users U_1 ,…, U_n and m

scientific papers P_1 ,…, P_m , the following is an example of a set of paper entries on which an inductive method can be applied.

{ < Agent , Jennings , 1997 , {P_2 , P_4 , P_9} , {U_1 , U_7} > ,
< Negotiation , Zlotkin , 1994 , { P_1 , P_9 , P_3, P_5} , { U_4, U_2} > ,
< AI , Maes , 1995 , {P_2} , { U_7, U_4, U_1} > }

The sets containing scientific papers are values of the collaborative attribute that characterises a user and the sets containing users are values of the collaborative attribute that characterises a paper. In this way, collaborative information, which is translated into attribute values, together with content-based information, which is also represented as attribute values, constitute the data to which an inductive learning method is applied to extract user preferences. It is important to note that the values of collaborative attributes can be very large sets when the numbers of users and items get large. This is a serious disadvantage of this integrated method.

6.4 Effectiveness of Preference Models

The effectivity of collaborative-based and content-based preference models depends on the applications they are used in. For example, collaborative-based preference models are effective for applications where either it is unrealistic to collect a large amount of information about the preferences of an individual user, or the number of users is too large. Using collaborative-based preference models is also effective for applications where the content of the items neither is available nor can be analysed automatically by a machine (e.g. items like a picture, video, sound, etc.). However, the collaborative-based preference models are less effective for applications like integrative negotiation (see [2],[15],[29]) in retail Electronic Commerce where negotiation is considered to be a decision making process over items that are described as multiple interdependent attributes. As explained, collaborative-based preference models are not defined in terms of attribute values and therefore they are less effective for applications like integrative negotiation.

On the other hand, content-based preference models are effective in applications where data are represented in terms of attribute values such that no more information than available is required. Also, content-based preference models provide sound results even in situations where there is only one single user. When a content-based preference model is constructed automatically, it will provide sound results if it is constructed on the basis of a set of rated items that is large enough. Note that collaborative-based preference models will fail to provide sound results in such a case. Content-based preference models are thus appropriate for applications like integrative negotiation since they are in terms of various attribute values. Moreover, since the preference model of a certain user is in terms of attribute values, a new items which is not rated by any other users can be decided to be interesting for that particular user. As we mentioned above, providing a preference model by a user in terms of various item's attributes is a time consuming activity. Therefore, we believe that in these applications user preferences should be modelled automatically. In order to achieve this goal in a multi-agent setting, we employ inductive logic programming which enables an agent to induce the preferences of a user in terms of item's attributes during its interactions with the user.

7 Automatic Preference Modeling with Inductive Logic Programming

Inductive logic programming (see [28]) lies at the intersection of machine learning and computational logic, as used in logic programming. It combines inductive machine learning with the representations of computational logic. Computational logic (a subset of first order logic) is a more powerful representation language than the classical attribute-value representation typically used in machine learning. This representational power is useful in the context of learning user preference models, because in this way more complex types of user preferences can be detected and described. Another advantage of inductive logic programming is that it enables the use of background knowledge (in the form of Prolog programs) in the induction process. Given the fact that DESIRE uses first order logic as knowledge representation formalism, this allows for an easy integration of both systems.

An ILP system takes as input examples and background knowledge and produces hypotheses as output. There are two common used ILP settings which differ in the representation of these data: learning from entailment ([11] compares different settings) and learning from interpretation (see [12]). We will use the second setting because of the time efficiency of this setting. In learning from interpretations, an example or observation of actions performed by the user — in this application requesting and rating a paper — can be viewed as a small relational database, consisting of a number of facts (such as 'author(Smith)' or 'interest(high)') that describe the specific properties of the example. An example may contain multiple facts about multiple relations. This contrasts with the attribute value representations where an example always corresponds to a single tuple for a single relation. We will show later that the extra flexibility gained with the learning from interpretations setting is very useful in user preference *modeling* (see Section 6).

The background knowledge takes the form of a Prolog program. Using this Prolog program, it is possible to derive additional properties from the examples. Let us illustrate this by showing how we can introduce a taxonomy using background knowledge:

```
topic(T) ← papertopic(T).
topic(T) ← isa(It,T), topic(It).
isa(agentsemantics,agent).
isa(agentarchitecture,agent).
isa(agent,artificial_intelligence).
```

This Prolog program recursively defines the topic-relation: a paper has topic T if either T is the topic directly related to the paper (through the papertopic-relation) or T is above It (defined with the isa/2 relation), an other topic related to the paper, in the taxonomy-lattice. By introducing the above background information the system adds to each example automatically all topic information: if we observe the user rating a paper with as topic agentarchitecture, the learning system adds that agentarchitecture, agent and artificial_intelligence are topics for this paper and will use this information when learning hypotheses.

There are two forms of induction: predictive and descriptive induction. Predictive induction starts from a set of classified examples and a background theory, and the aim is to induce a theory that will classify all the examples in the appropriate class. On the other hand, descriptive induction starts from a set of unclassified examples, and

aims at finding a set of regularities that hold for the examples. In the situation of a paper mediating agent, predictive induction could be used to predict the interest of a user in a paper. Descriptive induction on the other hand would try to find all regularities that hold in the set of examples, and would find rules such as:

If the author is Jennings and the user is interested, then the co-author is Wooldridge

Notice that these types of rules — although maybe revealing interesting and unknown knowledge — are not useful for predicting the user's interest. Since our aim is to predict the user's preference for unseen objects, we focus on predictive induction in the learning from interpretations setting, because in this system the focus is on finding rules useful for classification. This task can more formally be expressed as follows:

Given:

a set of classes C,

a set of classified examples E,

a background theory B

Find a hypothesis H such that:

for all $e \in E$, $H \wedge e \wedge B \models c$, and $H \wedge e \wedge B \not\models c'$

where c is the class of the example e and $c' \in C - \{c\}$.

To make the discussion more concrete we focus on one ILP system: Tilde (see [4],[5]). This system performs predictive induction in the learning from interpretation setting by inducing logical decision trees from classified examples and background theory. Consider for example the background knowledge that is mentioned above. Suppose also a set of observations describing papers and the interest of a user in those papers. In this application we let the user rate his interest in a paper on a score from 1 to 10, where higher numbers indicate a higher interest. We build user models for each user individually, so we collect all observations from a certain user in one file. As a result there is no need to add information about which user made these observations to the data that will be given to the learning system. The following is an example of one such observation. Notice that attributes (such as author) can have multiple values.

```
papertopic(agentarchitecture).
author('Jennings').
author('Mamdani').
aff('Jennings','Queen Mary & Westfield College').
aff('Mamdani','Imperial College').
interest(6).
```

We included the affiliation of the authors in the example. One could argue that such information can be stored in background knowledge. However the fact that an author can belong to different affiliations at the same time makes this a property which can't be computed from the other information in the example and hence can't be stored in background information.

Starting from the background knowledge and a set of observations Tilde can build hypotheses (represented as first-order logic decision trees) which predict the user's interest in a paper. The following is an example of such a hypothesis:

```
topic(agents) ?
+--yes: author(A), A = 'Jennings' ?
|     +--yes: author(B), aff(A,C), aff(B,D), C ✱ D ?
|     |     +--yes: interest(6)
```

```
|    |    +--no:  interest(4)
|    +--no:  interest(3)
+--no:  interest(1)
```

Tilde uses the standard induction tree algorithm for building this trees: look for a test which best separates the examples in sets belonging to the same class and repeat this procedure in each leaf of this tree until a stopping criteria is reached. Notice that this is a greedy approach: selecting the best splitting test at each level of the tree doesn't necessary result in the best global tree. Using look ahead, Tilde can make conjunctions of tests and use these as single tests, as clearly illustrated in the third line of the above tree.

The above hypothesis states that the user has interest 1 in papers that are not about agents. If it's a paper on agents written by Jennings the predicted interest value is 4 unless there is a co-author from a different affiliation, then the interest prediction is 6. Agent papers not written by Jennings have a predicted interest-value of 3. As this example shows, hypotheses can contain constants as well as variables.

Notice that this very simple example shows the power of inductive reasoning. From a set of specific facts, a general theory containing variables is induced. It is not the case that the induced theory deductively follows from the given examples. The Tilde system has the benefits (like most ILP systems) of being able to build complex hypotheses (using first order logic) and using background knowledge in finding these hypotheses. Moreover experiments have shown that the Tilde system scales nicely on large datasets (see [5]). More details of the Tilde system can be found in [4] and [5].

7.1 User Preference Modeling with Tilde

Following Section 6, we may define a user preferences model for a certain user u as a function φ_u mapping an observation o from the set of possible observations O onto a preference indicator value p from the set of possible preference indicator values P, a finite structured domain. Since Tilde can induce general hypotheses from specific observations and background knowledge, it can be used to induce φ_u automatically. This can be done by building a set of examples E, each example consisting of an observation o and $\varphi_u(o) = p$, the preference user u has for observation o. As the set of classes C we use the set of possible preference indicator values p from P. In this way we transform the construction of φ_u into a learning task. We can also add background knowledge B with information the system can use in constructing φ_u.

We illustrate this by an example. Consider again the task of building a preference model for a user who is looking for interesting scientific papers as discussed in Section 2. In this context each observation consists of information about a paper: title, author, year of publication, type of publication (journal, conference proceeding, workshop proceeding) and the topic. Attributes author and topic can be multi-valued. Other possible useful attributes are the affiliation of the author(s), the publishing company and the length of the article (number of words or pages). In background knowledge we put the general information that could help Tilde in constructing φ_u. As background knowledge we use an extension of the taxonomy on topics as introduced earlier in this section. However all other relevant information (e.g. background information on authors, publishers and affiliations) could be included as well. Finally

one has to create a language bias, specifying which concepts Tilde is allowed to use in constructing φ_u.

We conducted three types of experiments to answer three questions we had in mind: can the system detect complex rules, can the system detect rules if there is noise in the examples and can we construct user preference models from a small number of observations. For the first two experiments a set of 375 observations was used. All these experiments were performed on machine generated data. We produced 375 examples of paper descriptions and let the computer label them according to criteria we defined. These examples (but not the criteria used to label them) were then given to Tilde to learn user profiles. The reason for this type of experiments is to determine whether the agent would be able to find user patterns under the assumption that these exist. The question whether useful user patterns do exist is not answered by these experiments because it is application and user dependent.

For the first experiment we labelled all our examples using increasingly more complex rules:

1. If published before 1992 then interesting else not interesting.
2. If published before 1992 or if it is a journal paper then interesting else not.
3. If published before 1992 or if it is not a journal paper then interesting else not
4. If published before 1992 or if it is a journal paper or it is a paper by Jennings then interesting else not.
5. If published before 1992 or if it is a journal paper written in 1998 then interesting else not.

No background knowledge was used in these experiments. Each time we labelled all 375 examples according to one of the above rules and let the system learn on all these examples. We then inspect the user preference model produced to see if they match the rule used to label the examples. When using rule 1 no problems are encountered. Rule 2 adds a disjunction, but this easy for the system to learn because it's just an adding another branch. Rule 3 complicates rule 2 by using a negation of one of the tests. But this as well is easy for the system to learn because negating a test is the same as switching both branches at the node that corresponds with this test. Rule 4 extends rule 2 by adding another disjunction. In the example set there were only three examples of non-journal papers by Jennings published after 1991 but even then Tilde was able to find the correct user preference model. The fifth rule could be learned by Tilde, but only when look ahead (testing conjunctions of tests in stead of single tests) is enabled. Look ahead however increases the time to build the decision tree so when we test Tilde on this data set without look ahead it finds an user preference model which nearly matches the correct hypothesis: it has one test more, and misclassifies 1 out of the 375 examples.

Most of the time, a user doesn't behave completely according to rules. For instance, a user may be interested in Jennings' papers, but some of these papers he will rate as non interesting for different reasons. For a user preference *modeling* system to be useful, it should be able to cope with such 'noise' in the observations. We tested this by introducing noise in the experiment mentioned above. We labelled examples according to rule 1 and added 5%, 10% and 15% of noise (this means that such a portion of the examples were random labelled[1]). To test the preference models we

[1] interest is indicated by an integer between 1 and 10, were we used 1 for uninteresting and 10 for interesting. With random, we don't mean random either 1 or 10 but a random integer between 1 and 10

performed a cross validation: the dataset is divided in equal sets, all but one set is used to learn a preference model, which is then tested on the remaining set. This procedure is repeated with another set as testset until every set has once been used as testset. We performed this experiment first with rule 1. The system found user preference models that were as accurate as possible: respectively 95%, 90% and 85% accuracy. The system had learned in each case the correct preference model (and so would have a predictive accuracy of 100% if there is no noise in the testset). If we repeat this experiment with rule 3 the accuracies are comparable. The time to induce these user preference functions doesn't increase when the percentage of noise increases. However, if we perform this experiment with a rule that has a disjunction only supported by few observations (as in the author is Jennings branch of rule 4) we see that the accuracy on the testset remain the same but the system doesn't find a correct preference model. This occurs when there are more observations supporting random patterns created by the noise than the regular patterns. However, if the noise level is so high, can we consider this part of the user's preference?

In a final experiment we try Tilde to learn from few examples. Simple φ functions can be learned from as few as ten examples. When learning rule 4 in the first experiment we also noticed that, although only 3 out of the 375 examples supported this disjunction, the system was able to detect this and include it in the preference model. Tilde can be used to model the user preferences based on few examples and will build a simple model. When more observations become available that don't agree with the initial simple hypothesis, Tilde will construct a more complex hypothesis. From these experiments we can see that the Tilde system is able to learn simple user preference functions from few examples. Preference functions can be learned even when the observations are noisy. From previous experiments [32] we can conclude that in an attribute-value setting Tilde performs comparable with classic machine learning systems. In the next section we will elaborate more on the advantages and disadvantages of the ILP-approach compared with more classical machine learning algorithms in learning preference functions

7.2 Advantages and Disadvantages of ILP in User Preference Modeling

In the previous section we briefly illustrated the Tilde algorithm on a simple dataset. We will now introduce the specific ILP-features and illustrate how these are useful in user preference *modeling* by applying them to the above example.

Because in ILP examples are represented as a set of facts, it is easy to represent examples where attributes have multiple values. This is common in many electronic commerce applications: books can have multiple authors, songs can have multiple songwriters and performers, …, movies have multiple actors,… . If you want to represent such information in an attribute-value setting, you have to introduce multiple attributes of the same type (e.g. author1, author2, …). This however results in two problems: the number of these attributes has to be fixed in advance (e.g. maximum 5 authors) and attribute-value learners will take the order of the attributes into account (e.g. if author3 = Jennings then interesting) while in many applications this will be irrelevant. ILP systems can represent multi-valued attributes without these

disadvantages: facts can occur more than once and are unordered. We illustrated this already in the paper example.

One can also easily extend the learning task just by extending the examples. Let's illustrate this. The user preference model learned in previous examples was solely based on features of the paper itself, there was no use of the collaborative *modeling* approach as discussed in Section 6.2. However, extending the learning task to incorporate this collaborative *modeling* is very easy: each example still consists of the paper observations o (authors, title, type, …), but instead of one indication of the preference of one user, the example also contains the preference of all users who read this paper. An example could then look like this:

```
papertopic(agentarchitecture).
author('Jennings').
author('Mamdani').
aff('Jennings','Queen Mary & Westfield College').
aff('Mamdani','Imperial College').
interest(user12,6).
interest(user23,4).
interest(user76,4).
```

A possible resulting tree for user 23 could then be like this:

```
author('Jennings') ?
+--yes: interest(user12,A), A<5 ?
|       +--yes: interest(user23,8)
|       +--no:  interest(user23,4)
+--no:  interest(user23,1)
```

If it is a Jennings paper and user 12 has read it and had low interest in it, then the interest of user 23 will be 8, if user 12 had not read it or had an high interest in it, user 23 has an interest of 4. If it is no Jennings paper the interest of user 23 is 1. So the φ_u function is based on a mixture of properties of the paper and the interests of other users.

First notice that this collaborative *modeling* approach is more flexible than other collaborative approaches in which for user u users u' are identified such that $\varphi_u(o) = \varphi_{u'}(o)$ for many observations o, while in this approach we identify for user u users u' such that $\varphi_u(o) = f_{u'}(\varphi_{u'}(o))$ for observations o that obey to certain conditions (namely the tests higher up the decision tree), and $f_{u'}$ a function mapping the preferences of user u' onto the preferences of user u. Also notice that the above approach nicely merges the two approaches (collaborative and content based) into one integrated approach in a natural and easy way, due to the flexibility of inductive logic programming. Of course, in stead of information about specific users, information about clusters of users can be calculated using background knowledge (see next paragraph) and used (e.g. 'if the interest of the students in this paper is low then …').

It is very common in ILP to extend the dataset by introducing background knowledge, formulated in the form of static facts (e.g. situated_in('Imperial College','London').) or in the form of rules (e.g. situated_in(A,C) ← situated_in(A,B), situated_in(B,C)) which allow to infer new facts from the knowledge already available in the background knowledge and the example. We illustrated the use of background knowledge by introducing a taxonomy, but since Prolog is Turing complete, any computable information can be added to the example. This allows for easy integration of ILP systems with other systems. For instance, in the above

example we could use a clustering algorithm to find descriptions for clusters of users. This information could be added to the background knowledge of Tilde so the system could use the result of the clustering algorithm. Another example of the flexibility is the fact that integrating Tilde with DESIRE was very easy: background knowledge translated information in DESIRE representation into the representation in which the learning task was expressed.

ILP is based on logic programming, a declarative programming formalism. Due to its declarative nature, input as well as output of ILP systems are readable (for humans as well as for computers), in contrast to sub-symbolic systems like neural networks. This is a very important feature in the context of agents for electronic commerce because ILP user models can easily be translated to English sentences. In this way the user can check and understand his preference model the agent has built. Users will probably be more likely to delegate tasks to agents they can understand and check than to 'black box' agents.

As illustrated above, ILP has some advantages compared with other concept learning methods. Due to the use of background knowledge and the use of first order logic as representation language, ILP is especially suited in knowledge intensive learning tasks where the data is mainly symbolic. If there is only few or no background knowledge used and the observations can easily be expressed in an attribute-value representation, traditional concept learning algorithms such as C4.5 [31] will result in a comparable user model while these systems require less computing power. Although the ILP method can handle numeric attributes, it's mainly focussed on symbolic datasets. If the observations of the user are expressed as numbers (for example sensor readings (blood pressure, brain activity, …)) and the user preference model is a mathematical function of these readings, better techniques (such as neural networks) exist. Finally, because ILP systems search a larger space of possible solutions than other techniques, ILP systems require more computing power than most other techniques. Although ILP systems can handle large datasets [5], to our knowledge there doesn't exist a fast incremental ILP algorithm useful in time-critical applications.

So ILP systems have their limitations. But a broad range of electronic commerce applications deals with mainly symbolic data in an environment where useful background knowledge is common. ILP can not only build user preference models in such a setting, but is also able to provide this model in a representation that can easily be mapped on natural language, and in this way help the user understand and trust the system. Since electronic commerce applications are fairly new phenomena, these applications tend to change over time. ILP is a very flexible learning method which makes it easy to adapt the learning system to new situations. All these features makes ILP well suited to learn user preference models in electronic commerce applications.

8 Discussion

In this section, some of the recently developed and operational models of virtual market places and Web commerce based applications are briefly mentioned.

1. Kasbah

Kasbah (cf. [9], [10]) is a web-based multi-agent system using agents interacting with each other within the virtual market domain. The agents act on behalf of their users [9]. Price Negotiation is one of the interesting features applied within Kasbah [10].

2. Market Space

Market Space is an open agent-based market infrastructure. It is based on a decentralized infrastructure model in which both the humans and the machines can read information about the products and services, and everyone is able to announce interests to one another [13]. The aim in designing Market Space is to design a market place where searching, negotiation and deal settlement, e.g. interaction with users is done using agents. The AMP (Agent Marketplace Project) is a collaboration project between Uppsala University and Swedish telecom, Telia. Market Space has been developed mainly in Prolog. For the communication with the Web, the standard protocol (HTTP) has been used.

A difference with our approach is that these approaches have been implemented without using a principled design method, and do not use components as building blocks that are (formally) specified at a conceptual level. This is also a difference with the work described in [35]. The mediating agent architecture introduced here was designed and implemented in a principled manner, using the compositional development method for multi-agent systems DESIRE [6]. Due to its compositional structure it supports reuse; a flexible, easily adaptable architecture results.

Required properties or functionalities of agents can be formalised, and the relation between required properties and underlying assumptions can be established in a formal manner. An example of a result of such a formal analysis is the relation between basic functionalities (required properties) and available knowledge (assumptions) discussed in Section 5 (see *Figure* 2). In this paper the result of formal analysis was used in the agent model; the formal analysis itself was done by us as designers. To support this, a compositional verification method for multi-agent systems has been developed and successfully applied to verify the behaviour of a multi-agent system for one-to-many negotiation (see [7]), and to give a formal analysis of pro-activeness and reactiveness (see [20]). One of the more ambitious aims of our future research is to explore possibilities to include these formal analyses themselves in an agent model, and not only the results obtained by them.

On the basis of the above discussion of techniques to construct preference *modeling*, the following claims can be made. A proper approach for preference *modeling* in a multi-agent setting should:

1. Allow agents to induce preferences of the involved participants automatically by observing their behaviour.
2. Be capable of handling the changes in the interests of participants that take place over the time by adjusting their preference models accordingly.
3. Be robust with respect to the partiality of information about preferences.
4. Allow for re-use of a preference model in different domains and for different purposes.

Note that none of the approaches mentioned in Section 6 can handle the second aspect real time, i.e., without computing the whole preference model over again. Likewise, the ILP method introduced in Section 7 cannot handle this problem real time. The design in this study is such that the mediating agent applies the ILP method

when enough new observations have been made. Furthermore, the approaches that are mentioned in Section 6 neither support the fourth facility. This facility may be realised by defining compositional preference models, i.e., various primitive preference models that can be composed to each other to form the preference model of a user. Each primitive preference model can then be reused in various configurations and thus for different applications. We will address this in more details in future studies.

Acknowledgements

Mehrzad Kharami and Pascal van Eck supported experiments with some early instantiations of the mediating agent model. This work was partially supported by the NWO project "Automatic and Dynamic Configuration of a Multi-agent system for Electronic Commerce, 612-060-004". Nico Jacobs is financed by a specialisation grant of the Flemish Institute for the promotion of scientific and technological research in the industry (IWT).

References

BASU, C. , HIRSH, H. , COHEN, W. Recommendation as Classification: Using Social and Content-Based Information in Recommendation. Proceedings of the fifteenth National Conference on Artificial Intelligence (pp. 714-720), 1998.

BENN, W. , GÖRLITZ, O , NEUBERT, R. Enabling Integrative Negotiations by Adaptive Software Agents. In Klusch, O. Shehory, and G. Weiss (eds.), Cooperative Information Agents III, Proceedings of the Third International Workshop on Cooperative Information Agents, CIA'99. Lecture Notes in Artificial Intelligence, vol. 1652. Springer Verlag. 1999.

BILLSUS, D. , PAZZANI, M. Learning Collaborative Information Filters. Proceedings of the Fifteenth International Conference on Machine Learning (pp. 46-54). Madison, WI: Morgan Kaufmann, 1998.

BLOCKEEL, H. , De RAEDT, L. Lookahead and discretization in ILP. In Proceedings of the 7th International Workshop on Inductive Logic Programming, volume 1297 of Lecture Notes in Artificial Intelligence, pages 77-85. Springer-Verlag, 1997.

BLOCKEEL, H. , De RAEDT, L. , JACOBS, N. , DEMOEN, B. Scaling up inductive logic programming by learning from interpretations. Data Mining and Knowledge Discovery, 3(1):59-93, 1999.

BRAZIER, F.M.T., DUNIN-KEPLICZ, B., JENNINGS, N.R., TREUR, J., Formal specification of Multi-Agent Systems: a real-world case. In: V. Lesser (ed.), Proceedings of the First International Conference on Multi-Agent Systems, ICMAS'95, MIT Press, Cambridge, MA, 1995, pp. 25-32. Extended version in: International Journal of Cooperative Information Systems, M. Huhns, M. Singh, (eds.), special issue on Formal Methods in Cooperative Information Systems: Multi-Agent Systems, vol. 6, 1997, pp. 67-94.

BRAZIER, F.M.T., CORNELISSEN, F. GUSTAVSSON, R., JONKER, C.M., LINDEBERG, O., POLAK, B., TREUR, J., Compositional Design and Verification of a Multi-Agent System for One-to-Many Negotiation. In: Proceedings of the Third International Conference on Multi-Agent Systems, ICMAS'98, IEEE Computer Society Press, 1998.

BRAZIER, F.M.T., JONKER, C.M., TREUR, J., Formalisation of a cooperation model based on joint intentions. In: J.P. Müller, M.J. Wooldridge, N.R. Jennings (eds.), Intelligent Agents III (Proc. of the Third International Workshop on Agent Theories, Architectures and Languages, ATAL'96), Lecture Notes in AI, volume 1193, Springer Verlag, 1997, pp. 141-155.

CHAVEZ, A., MAES, P., Kasbah: An Agent Marketplace for Buying and Selling goods. In: Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, PAAM'96, The Practical Application Company Ltd, Blackpool, 1996, pp. 75-90.

CHAVEZ, A., DREILINGER, D., GUTMAN, R., MAES, P., A Real-Life Experiment in Creating an Agent Market Place. In: Proceedings of the Second International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, PAAM'97, The Practical Application Company Ltd, Blackpool, 1997, pp. 159-178.

De RAEDT, L. Logical settings for concept learning. Artificial Intelligence, 95:187-201, 1997.

De RAEDT, L. , DZEROSKI, S. First order jk-clausal theories are PAC-learnable. Artificial Intelligence, 70:375-392, 1994.

Erikson, J. Finn, N., Market Space: an open agent-based market infrastructure. Master's Thesis, Computer Science Department, Uppsala University, Sweden, 1997.

Eugene, C. , Freuder and Richard J. Wallace. Suggestion Strategies for Constraint-Based Matchmaker Agents. In Principles and Practice of Constraint Programming - CP'98. Pisa, October, 1998. http://www.cs.unh.edu/ccc/papers1998.shtml.

GUTTMAN, R.H. , MAES, P. Agent-mediated Integrative Negotiation for Retail Electronic Commerce. Proceedings of the Workshop on Agent Mediated Electronic Trading (AMET'98), Minneapolis, Minneasota, April 9, 1998.

GUTTMAN, R.H. , MOUKAS, A.G. , MAES, P. Agent-mediated Electronic Commerce: A Survey. Knowledge Engineering Review, June 1998.

HILL, W. , STEAD, L. ROSENSTEIN, M. FURNAS, G. Recommending and Evaluating Choices in a Virtual Community of Use. Proceedings of the Conference on Human Factors in Computing Systems (CHI95), pp 194-201, Denver, CO, ACM Press, 1995.

HOFMANN, T. , PUZICHA, J. Latent Class Models for Collaborative Filtering. Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sweden, July 31- August 6, 1999.

JONKER, C.M., TREUR, J., Compositional Design and Maintenance of Broker Agents. Technical Report, Vrije Universiteit Amsterdam, Department of Mathematics and Computer Science, 1998.

JONKER, C.M., TREUR, J., Compositional verification of multi-agent systems: a formal analysis of pro-activeness and reactiveness. In: W.P. de Roever, H. Langmaack, A. Pnueli (eds.), Proceedings of the International Workshop on Compositionality, COMPOS'97, Springer Verlag, 1998.

KEENEY, R. , RAIFFA, H. Decisions with Multiple Objectives: Preferences and Value Trade-offs. Jhon Wiley and Sons, 1976.

KUOKKA, D., HARADA, L., On Using KQML for Matchmaking. In: V. Lesser (ed.), Proceedings of the First International Conference on Multi-Agent Systems, ICMAS'95, MIT Press, Cambridge, MA, 1995, pp. 239-245.

LANG, K. Learning to filter news. Proceedings of the twelfth International Conference on Machine Learning, Lake Tahoe, CA, 1995, (pp 331-339).

LANGLEY, P. Machine Learning for Adaptive User Interfaces. Proceedings of the 21st German Annual Conference on Artificial Intelligence (pp. 53-62). Freiburg, Germany: Springer, 1997.

LANGLEY, P. User Modeling in adaptive interfaces. Proceedings of the seventh International Conference on User Modeling, 1999.

LIEBERMAN, H. Letizia: An Agent that Assists Web Browsing. Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95). Montreal, Canada, 1995.

MARTIN, D., MORAN, D., OOHAMA, H., CHEYER, A., Information Brokering in an Agent Architecture. In: Proceedings of the Second International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, PAAM'97, The Practical Application Company Ltd, Blackpool, 1997, pp. 467-486.

MUGGLETON, S. , PAGE, C.D. A learnability model for universal representations. In S. Wrobel, editor, Proceedings of the 4th International Workshop on Inductive Logic Programming, pages 139-160, Sankt Augustin, Germany, 1994. GMD.

OLIVEIRA, E. , FONSECA, J.M. , STEIGER-GARÇÃO, A. Multi-criteria negotiation on Multi-Agent Systems. The First International Workshop of Central and Eastern Europe on Multi-agent Systems (CEEMAS'99) International Conference , St. Petersburgh, June 1999.

PAZZANI, M. , BILLSUS, D. Learning and Revising User Profiles: The identification of interesting web sites. Machine Learning 27, pp 313-331, (1997).

Quinlan, J. R. , Programs for Machine Learning. Morgan Kaufmann, 1992.

Roberts, S. and Jacobs, N. and Van Laer, W. and Muggleton, S. and Broughton, J. A Comparision of ILP and Propositional Systems on Propositional Traffic Data. In: Proceedings of the 8th International Conference on Inductive Logic Programming, lecture notes on AI nr 1446, Springer verlag, 1998.

SANDHOLM, T., LESSER, V., Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Network. In: V. Lesser (ed.), Proceedings of the First International Conference on Multi-Agent Systems, ICMAS'95, MIT Press, Cambridge, MA, 1995, pp. 328-335.

SHARADANAND, U. , MAES, P. Social Information Filtering: Algorithms for Automating "Word of Mouth". Proceedings of the CHI-95 conference, 1995.

TSVETOVATYY, M., GINI, M., Toward a Virtual Marketplace: Architectures and Strategies. In: Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, PAAM'96, The Practical Application Company Ltd, Blackpool, 1996, pp. 597-613.

VINCKE, P. Multicriteria Decision-aid. John Wiley and Sons, 1992.

WOOLDRIDGE, M., JENNINGS, N.R., Agent theories, architectures, and languages: a survey. In: [38], pp. 1-39.
WOOLDRIDGE, M., JENNINGS, N.R. (eds.), Intelligent Agents, Proc. of the First International Workshop on Agent Theories, Architectures and Languages, ATAL'94, Lecture Notes in AI, vol. 890, Springer Verlag, 1995.

Amazon URL: http://www.amazon.com
AuctionBot URL:<http://auction.eecs.umich.edu>
BargainFinder URL:<http://bf.cstar.ac.com>
eBay URL:<http://www.ebay.com>
FireFly URL:<http://www.firefly.com>
Jango URL:<http://www.jango.com>
MarketMaker URL:<http://maker.media.mit.edu>
MovieFinder URL:<http://www.moviefinder.com>