International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:7, 2007

A Learning Agent for Knowledge Extraction from an Active Semantic Network

Simon Thiel, Stavros Dalakakis, and Dieter Roller

***Abstract*—**This paper outlines the development of a learning retrieval agent. Task of this agent is to extract knowledge of the Active Semantic Network in respect to user-requests. Based on a reinforcement learning approach, the agent learns to interpret the user's intention. Especially, the learning algorithm focuses on the retrieval of complex long distant relations. Increasing its learnt knowledge with every request-result-evaluation sequence, the agent enhances his capability in finding the intended information.

***Keywords*—**Reinforcement learning, learning retrieval agent, search in semantic networks.

I. Introduction

IN these days the speed of development is critical for the success of any enterprise. Rapid Product Development (RPD) is an up to date answer for many companies. A knowledge representation independent from application and system can give an extra speedup for complex development processes and the required communication. As an essential feature of such a system it is not only necessary to insert the data but to find the demanded information in a reliable way. Therefore we developed a learning retrieval agent (LR-Agent) that supports the user in searching the knowledge representation. The learning aspect regards the users view in his requests and translates those requests into the given structure and content of the system.

The Basic Framework

The Active Semantic Network (ASN) is designed to provide a global and central knowledge base and information structure for enterprises to represent the rapid prototyping process and the knowledge around the product. On the one hand the ASN can handle data of different applications, enrich them with meta information and provide status and process control on the other hand all kind of information can be modeled primary as a combination of ASN concepts and relations. While the first mentioned data does not provide any semantic information about the RPD domain, however the semantics of ASN concepts and relations can be given in different ways, but their interpretation is not absolutely determined and has to be interpreted by human or artificial intelligence methods. Since facts and their semantics can be represented in several ways leading to a certain complexity adding more facts blows up the ASN to a large network. A search engine that regards semantically information becomes indispensable.

Choosing the semantic net approach for representing knowledge about RPD, we try to solve problems related rather to the domain then to knowledge representation field. On the other hand retrieving knowledge depends from the kind of knowledge representation formalisms. So, knowledge representation (KR) so far represented knowledge in a high abstract form without closer specification of the domain. Respectively the ways of retrieving knowledge was adequate to the representation. A KR considering the domain is an ontology and for that the retrieval opportunities are ad hoc methods. Expressing the RPD domain by defining ontologies means that implicit knowledge could be reasoned. This limits the tasks of retrieving knowledge on the explicit represented knowledge and causes retrieval questions to be moved into the problem solving space of request formulation.

Complex systems like RPD elaborate intelligence solutions using the autonomous behaviour of multi agent system (MAS). The chosen architecture is based on the Client-Server technology. Interagent communication is based on the multicast principle. After closer examination of the requirements of RPD we realized five different types of Agents [1], [2]. These are parts of the MAS and are generated as necessary. Main criterion for their architecture was a design by services expressing the user's view.

The five agent types are:

1) Monitoring Agent: monitors the ASN and notifying changes.
2) Coordination Agent: supports coordination within the RPD by finite state machine.
3) Transaction Agent: supports transactions protected processes and transaction protected execution of other agents within the MAS.
4) Aggregation Agent: prepars in appropriate format retrieved knowledge.
5) Retrieval Agent: retrieves ASN knowledge.

In our aim to search for knowledge with more intelligent tools we involved the learning retrieval agent.

Manuscript received July 15, 2005. This work was supported by the Deutsche Forschungsgemeinschaft (German Research Foundation) in the project Sfb374.

S. Thiel is with the Institute of Computer-aided Product Development Systems, University of Stuttgart, Germany. (e-mail: simon.thiel@gmx.de).

S. Dalakakis is with the Institute of Computer-aided Product Development Systems, University of Stuttgart, Germany. (e-mail: dalakakis@uni-stuttgart.de).

D. Roller is with the Institute of Computer-aided Product Development Systems, University of Stuttgart, Germany. (e-mail: roller@uni-stuttgart.de).

The design of the ASN specifies a layer of server agents to access the stored data. Therefore the claimed search engine is realized as a specialized search agent. A good search agent needs to model an interpretation of the semantics of the ASN that corresponds with the user view of the ASN. Thus a request can be handled in the way that is meant by the user. The only method to generate such a model of the user view/interpretation is continuous learning which results are rated good for a certain request. The described learning algorithm is of the kind Reinforcement Learning. Its main task is to classify user requests according to their interpretation. This enables a generalized learning, that doesn't only regard specialized requests but request classes with appropriate interpretation.

This paper describes the developed LR-Agent with particular consideration of the user request classification.

II. Motivation

Before focusing on the problems motivating the usage of a *learning* retrieval agent, we present a short overview of the user agent interaction process in the retrieval situation without a learning component.

A. Interaction Process

The sequence starts with the user posing a request to the agent. The agent retrieves information from the ASN and returns the result to the user. The user may now be happy with the received information. If he isn't, he will modify the request according to an assumpted functionality of the agent hoping for a better result in the second run. Thus the user tries to learn the behavior of the agent. This is comprehensible observing any Google user.

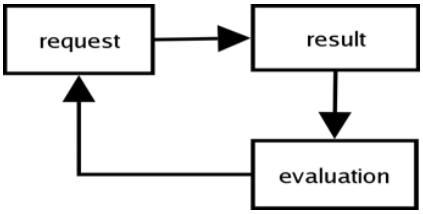

Fig. 1 User – Agent: interaction process

The idea of the LR-Agent is to add learning capabilities to the agent's side of communication. This tends to a sensitive interaction between user and agent. The agent learns to memorize the user's intention. This means remember request sequences and their finally result and derive a retrieval method for further comparable requests. To afford that the agent needs information about which results have been helpful for the user. Therefore we added an evaluation step into the interaction process. As illustrated in figure 1.

B. Request

The input of the LR-Agent is the user request. Three different kinds of requests can be classified. These are denoted in the table 1.

A *path* is defined as a sequence, beginning with a concept called "start", followed by relation-concept pairs. Which means that, beginning at "start", every following concept is connected to its predecessor by the given relation. This tends to a string like: "concept – relation – concept - ... - relation-concept".

TABLE I
Classification of Requests

Type	Request for	Result
1	Single concepts	Set of concepts
2	Relations	Set of concept pairs
3	Long distance relations	Set of paths

While it is easy to find algorithms to answer the first and second request types, finding fitting results containing *long distant relations* is much more difficult. This is because the semantics of those relations is not designed explicit and has to be derived by the semantics of relations and concepts located on the route between "start" and "target". An automatically derivation of such semantics is only possible if every potential kind of relation is explicit defined and what is more, semantics for every possible combination of relations have to be computable. Since these preconditions are generally not satisfied, the interpretation of long distant relations remains a human task.

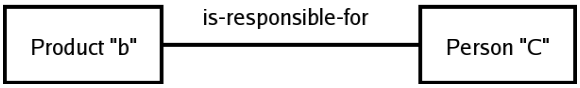

Fig. 2 Relation: User's view

Although the user may be able to interpret complex relations in a just way he doesn't have an actual and complete view of the ASN. Because of its complexity even the designer may be unable to keep track of the ASN as a whole. Since the user does not know the exact design of the ASN, his assumed view of the modeled knowledge will very likely differ from the real design, which poses a problem even for users asking *type two* questions, according to table 1.

C. Example

Suppose that a user wants to know who is responsible for the developement of product "b". He will request "product b" as "start" concept and "is-responsible-for" as the relation, that leads to the demanded target. Obviously the user imagines an ASN design as shown in figure 2. Though the actual ASN design may not have an "is-responsible-for" relation at all, but the same semantics could be represented by a structure as shown in figure 3. It is the task of the learning retrieval-agent to map the user's view of the ASN to a search algorithm working on the real structure of the ASN, thus retrieving results satisfying the user's intention.

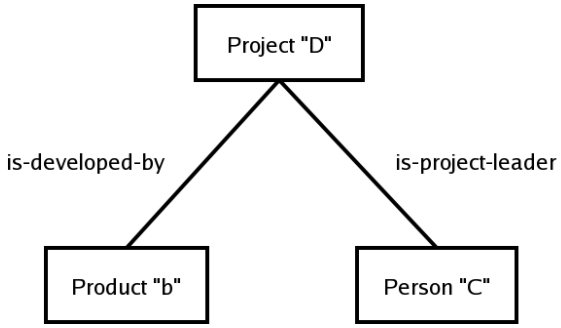

Fig. 3 Relation: ASN design

D. Learning Methods

Learning in this context is regarded as part of the communication process between user and agent. As shown in figure 1 this process consists of three steps: request, result and evaluation. In the agents view these steps represent the interface to the user.

In [3],[4],[5] two different learning methods are distinguished: Learning from examples and Reinforcement Learning (RL). The latter means *learning by experience* which is not restricted to a temporarily terminated learning phase, but can be continued for the whole operation time. Thus RL provides an appropriate solution for the requested features: Adapting of the agents behavior to the alternating ASN model and the mutating user reaction as a consequence of the communication with the agent [6].

III. Approach

However the LR-Agent, which can be seen on figure 4, is considered to learn the user's intention with the restricted view of the actual request. That means to interpret each request and map it to an adequate search method.

A. Preliminary Considerations

This interpretation of user requests raises two questions:

- How can the request space be *classified* for comparable requests?
- In which way can the *interpretation of user requests* be learned by the agent from the user?

Concerning the first question, the classification of user requests is limited by the number of search algorithms leading to different results. Furthermore, it would be reasonable to handle analog requests with the same search algorithm. The search algorithm assigned to a class of user requests transforms the user's view of the ASN to its real structure and content. Therefore it will be called in the following, *interpretation* of request.

B. Requests

For the LR-Agent we decide to restrict the request formalism to a fix request structure. For ergonomic purpose a natural language interface would be better, but it involves the trouble of natural language processing, which includes as a sub problem, the problem of understanding the user's intention, called as pragmatics. Therefore the request interface provides concept specifications for start and target and the specification of the linking relation. This structure is expressed in a DTD defined XML file. Therefore user requests can easily be submitted by the communication protocol of the given agent framework. To show the functionality we built a GUI prototype to express requests in an easy way.

C. Characteristic

The classification of user requests is done by a profile, called *characteristic*, assigned to each interpretation. The characteristic describes the significant properties of requests sharing the same interpretation. Characteristics are generated out of former generalized requests. They are modified in the learning process adapted to the changing user behavior and state of the knowledge base. In this context, knowledge base means, a list of such request characteristics each assigned to a structure called request *interpretation.*

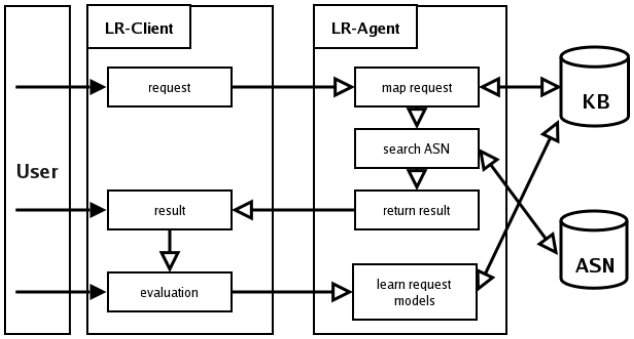

Fig. 4 LR-Agent: Accessing dependencies

D. Interpretation

The interpretation of a request means a description how to adapt the search algorithm in respect to the concrete assigned user request. Therefore the interpretation is separated in two parts. The first part is called *relation processor*; it defines which ASN relations should be followed to find long distant relations. This implies to have a set of start concepts, from which the algorithm begins to search. Those start concepts are found with regard to the criteria given by the user. In some cases it can be useful to add fix start concepts to an interpretation. This is done for concepts that have proven relevant in the context of an interpretation. Such interpretations are assigned to their fitting characteristic. Both will further on be called a request model.

E. Knowledge Base

A knowledge base (KB) contains a list of request models, which consists of a characteristic and an interpretation. This KB module as shown in figure 4, represents the information needed for the mapping "user view" on "ASN model".

The KB may be excluded from the LR-Agent which leads to a bunch of interesting problems regarding the multi agent

framework. These are related with synchronization of various agent instances sharing a single KB. Other aspects apply to problems regarding the KB administration which needs special algorithms to delete irrelevant request models and join equal ones. But in this paper we will not further discuss them.

F. Learning Techniques

Technically, it is the knowledge base that represents the learnt data. Therefore learning techniques for the aspects, characteristic and interpretation have to be found. The learning process depends on former information and the three components of the particular request: The request, the result and the evaluation of the result. While the request and the result are important to deliver boundary conditions the evaluation provides the information to consider the quality of the assignment on the one hand and that of the used interpretation on the other.

While results can be represented as a list of paths, an evaluation can simply be given by rating the result paths. The prototype client provides a user review of the relevance of each result path on a scale between zero and ten. Then the evaluation is sent back to the LR-Agent as shown in figure 2.

The learning algorithm for the interpretation part of the request model regards every single path and readjusts the relation processor according to the relations along the path and the paths rating. The characteristic part regards a global evaluation that is computed out of the single path evaluations. The learning algorithm follows the idea to generalize the characteristic if the "request – request model" assignment has proven successful and to specify it otherwise.

IV. Conclusions

The challenge to learn the user's view of the ASN requires the classification of the request space. In our approach this is done by request models, where the characteristic part assigns each request to an interpretation. The used request models are summarized in the agent's knowledge base, which helps to interpret further requests. The interpretation steers the searching process on the ASN.

The biggest benefit from our approach is the ability to provide a flexible and independet retrieval method. The user is supported in the searching process. The LR-Agent was implemented as a learning communications counterpart that has the ability to memorize previously posed requests and even more, derives and assigns similar requests.

References

[1] Dalakakis, S., Stoyanov, E., Roller, D.: A Retrieval Agent Architecture for Rapid Product Development. In: Perspectives from Europe and Asia on Engineering Design and Manufacture, EASED 2004, X.-T. Yan, Ch-Y. Jiang, N. P. Juster, (eds.), Kluwer Academic Publishers, 2004, pp. 41-58.

[2] Dalakakis, S.; Diederich, M.; Roller, D.; Warschat, J.: Multiagentensystem zur Wissenskommunikation im Bereich der Produktentstehung–Rapid Product Development. In: Wirtschaftsinformatik 2005 / Ferstl, O. K.; Sinz, E. J.; Eckert, S.; Isselhorst, T. (Eds.). Heidelberg: Physica-Verlag, 2005. ISBN 3-7908-1574-8 pp. 1621–1640.

[3] Stuart J. Russell and Peter Norvig. Artificial Intelligence: a modern approach. Prentice Hall, 1995.

[4] Richard S. Sutton und Andrew G. Barto. Reinforcement Learning: An Introduction. Bradford Books, 1998.

[5] Tom M. Mitchel. Machine Learning. McGraw-Hill, 1997.

[6] I. Kreuz, D. Roller: Reinforcement Learning and Forgetting for knowledge based Configuration, Artificial Intelligence and Computer Science, 83-121, Nova Science Publishers, Inc., 2005.