

An Adaptive Plan-Based Dialogue Agent: Integrating Learning into a BDI Architecture

Anh Nguyen
School of Computer Science and Engineering
University of New South Wales
Sydney NSW 2052, Australia
anht@cse.unsw.edu.au

Wayne Wobcke
School of Computer Science and Engineering
University of New South Wales
Sydney NSW 2052, Australia
wobcke@cse.unsw.edu.au

ABSTRACT

We consider the problem of dialogue adaptation in our Smart Personal Assistant (SPA), which uses a plan-based dialogue model. We present a novel way of integrating learning into a BDI architecture so the agent can learn to select the most suitable plan amongst those applicable, enabling the SPA to tailor its responses according to the conversational context and the user's physical context, device and preferences.

Categories and Subject Descriptors

I.2.1 [Artificial Intelligence]: Applications and Expert Systems—*natural language interfaces*; I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence—*intelligent agents*

General Terms

Human Factors

Keywords

BDI agents, dialogue management, adaptive dialogue systems, symbolic machine learning

1. INTRODUCTION

The Smart Personal Assistant (SPA) [9] is an application for desktops and PDAs that allows users to conduct spoken dialogues with the system to remotely access and manipulate their e-mail and calendar. The SPA consists of a dialogue manager agent and a collection of task assistants, each specializing in a particular task domain. The dialogue manager [5] is implemented using a BDI agent architecture, which has a set of plans for processing user requests. This BDI approach allows a high level of abstraction in designing the dialogue model, which consists of complex but modular plans, each associated with a communicative goal or dialogue aspect. There are also plans for handling

domain-specific tasks. The control flow of the dialogue is not specified directly, but derived automatically using the plan selection mechanism as part of the BDI interpretation cycle. This approach preserves modularity within the dialogue model at the level of plans, allowing domain-independent discourse-level plans to be reused for other domains. The belief state of the dialogue manager contains the past and current states of the dialogue as a stack of conversational acts performed by the user, the dialogue manager and other task assistants. These conversational acts constitute a hierarchy of dialogue segments, each having a segment purpose which represents a task-specific intention of the user.

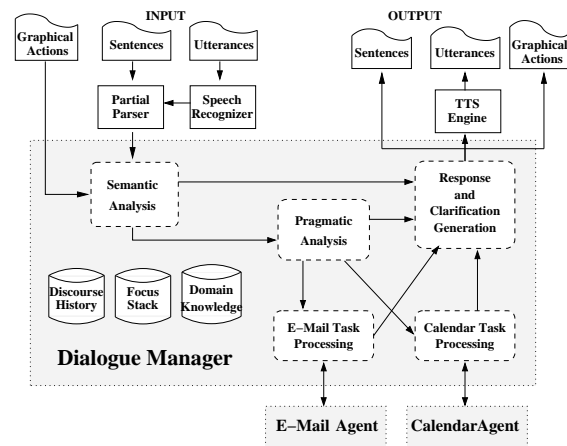


Figure 1: Dialogue manager's plans and beliefs

The dialogue manager has about forty plans, which can be roughly grouped into five classes according to their function, as in Figure 1. The first group, *semantic analysis*, is for classifying the user's requests into a task domain and formulating its semantic representation using the agent's linguistic knowledge. The second group, *pragmatic analysis*, is for determining the user's underlying conversational act and associated task-specific intention, which may involve resolving references. The *e-mail* and *calendar task processing* groups contain domain-specific plans for performing domain tasks, which requires interaction with the other task assistants. The last group is for generating responses to the user, including system requests for clarification.

In this paper, we consider the problem of dialogue adaptation in the SPA. The motivation is that the usability of the SPA on a mobile device with small screen and limited band-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

AAMAS'06 May 8–12 2006, Hakodate, Hokkaido, Japan.
Copyright 2006 ACM 1-59593-303-4/06/0005 ...\$5.00.

width can be enhanced by adaptively tailoring responses according to the conversational context and the user’s physical context, device and preferences. For example, e-mail messages are often too long to fit on one PDA screen, thus showing only a message summary could sometimes be more desirable. Similarly, in different conversational contexts, the same information might be presented to the user in different ways, e.g. using a different modality, format or layout. Because the SPA’s dialogue manager uses plans for generating responses, the problem of dialogue adaptation becomes the problem of learning which dialogue plan should be selected and executed in a given context.

There has been comparatively little work in the area of dialogue adaptation; existing work tends to be concerned with either adapting the system’s questions for eliciting user preferences [7] or learning strategies in a game-theoretic approach to dialogue modelling such as the Email Voice Interactive System (ELVIS) by Walker *et al.* [8] and the work by Jokinen *et al.* in the Interact project [3]. In the game-theoretic approach, training dialogues are used to build an adaptivity model using reinforcement learning. However, the dialogue models are less complex than that of the SPA.

There have been very few attempts to incorporate learning within BDI models of agency [1, 6], of which none has looked at the problem of how the agent learns to select the most appropriate plan amongst those applicable in a given context. Typically, this work provides a loose coupling of the agent architecture and the learner, while for the problem of learning plan selection, a closer integration is required.

2. AN ADAPTIVE DIALOGUE AGENT

To build an adaptive plan-based dialogue manager, there are two problems to be solved: (i) defining the possible system actions in any given situation, and (ii) defining the mechanism for the system to decide when to choose which action. Using the plan-based approach to dialogue management, system modularity is facilitated through the use of plans, each handling some dialogue aspect or domain task. Hence the possible choices of system action can be modelled as different plans applicable in the same situation, and the learner is used for the selection mechanism to choose the most suitable plan according to the agent’s beliefs, which represent the user’s context and/or preferences. Thus the learner accepts as input a set of possible plans and some of the agent’s beliefs and outputs the most suitable plan(s).

The dialogue manager of the SPA is implemented using a Java-based PRS-style agent platform, JACK Intelligent Agents™ [2]. JACK provides meta-level reasoning, posting a special `PlanChoice` event, as a way to overwrite the agent’s plan selection mechanism, making the integration with learning easier. We have chosen to integrate the Alkemy symbolic decision tree learner [4] into the dialogue manager to develop an adaptive dialogue agent. In Alkemy, the learning individuals are represented in a typed, higher-order logic, which supports representation of domain data with complex structure. Moreover, Alkemy provides a highly-expressive predicate rewrite system to constrain the hypothesis space for the learner.

Figure 2 shows the learning problem for the meta plan to decide which plan to use to display a set of e-mail messages to the user. Decision tree learners are not able to learn a set of values, and in our learning problem it is necessary to predict more than one possible plan, so given the current

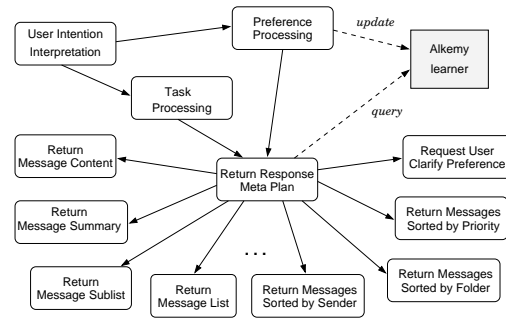


Figure 2: Example learning plan selection

context, for each available `Return Response` plan, a learning individual is generated that includes information about the current context and the plan name. This is classified as either `True` or `False`. If the result is `True`, the plan is considered an option; otherwise, the plan does not meet the user’s preferences is not an option. If there is more than one plan classified `True`, they are all options, and in this case the `Request User Clarify Preference` plan is executed, asking the user to specify the preferred system action (i.e. the most preferred plan) and then updating the learner accordingly. The Alkemy specification for this learning problem includes the following data constructors:

```
PDA, Desktop : Device ;
Speech, Text, None : Mode ;
Search, Read, Show, ..., Notify : Task ;
Low, High, Normal : Priority ;
Length = Int ;
Folder = String ;
ReturnMsgList, ..., ReturnMsgSortedBySender : Plan ;
True, False : Class ;
```

The `Mode` element indicates the current modality of the dialogue, where `Text` corresponds to typed text and `None` means the user is currently using only the GUI. The dialogue result included in the learning individual includes a set of e-mail messages. Each message is represented by four components: the sender, the length, the folder and the priority. Thus each learning individual (training example) is of the `Product` type defined as follows:

$$Ind = Device \times Task \times Mode \times (Set\ Email) \times Plan;$$

Each learning example is pre-computed and sent as input to the learner. The individuals include information about the current context, e.g. the modality of interaction, the device, etc., the result set of e-mail messages and the name of the plan to be considered. The learning problem is to predict whether a plan is an option in the current context according to the user’s preferences. Thus the function that needs to be learned is $ResponsePlan : Ind \rightarrow Class$.

Alkemy uses a predicate rewrite system for constructing the hypothesis space. Predicates are composed from basic functions called transformations. For example, a transformation on `Email` to extract its length is a projection of its second component as follows:

```
projLength : Email  $\rightarrow$  Int;
projLength : project(1);
```

The following predicate is true iff the e-mail set contains exactly one message whose length is less than thirty lines:

```
and (projEmails  $\circ$  numOfEmails(true)  $\circ$  eq1)
(projEmails  $\circ$  setEmailExists(projLength  $\circ$  lt30))
```

A part of the predicate rewrite system for our learning

problem is shown in Figure 3. The predicates are constructed by applying the predicate rewrites to expand the predicate tree starting from the root. For example, replacing *top* in (1) by (7) results in a predicate which is true iff the user’s device is a PDA. Predicate rewrite (3) is useful when there is no e-mail message in the result. Expanding (4) or (5) results in predicates for checking the length or the priority of the only e-mail message in the result. The rule (6) is used for constraining the total number of messages. Note that defining this predicate rewrite system requires the expression of domain-specific knowledge. Furthermore, constraining the learner’s hypothesis space is necessary because of the limited availability of training data.

```

(1) top  $\mapsto$  projDevice.top;
(2) top  $\mapsto$  projPlan.top;
(3) top  $\mapsto$  and (projEmails.numOfEmails(true).eq0)(top)
(4) top  $\mapsto$  and (projEmails.numOfEmails(true).eq1
                (projEmails.setEmailExists1(projLength.top)));
(5) top  $\mapsto$  and (projEmails.numOfEmails(true).eq1
                (projEmails.setEmailExists1
                 (projPriority.top)));
(6) top  $\mapsto$  projEmails.numOfEmails(true).top;
(7) top  $\mapsto$  eqDevicePDA;
(8) top  $\mapsto$  eqPriorityHIGH;
(9) top  $\mapsto$  lt30;

```

Figure 3: The predicate rewrite system

We first show an example dialogue without adaptation:

```

User  Is there any new mail from Wayne?
SPA   You have one new message from Wayne Wobcke.
SPA   Displays the message from Wayne Wobcke.
User  Find all messages about meeting in the Inbox.
SPA   There are twenty messages about meeting in your Inbox.
      Displays the headers of the twenty messages.
User  Show me the one from John.
SPA   Here you are.
      Shows the content of the message from John Lloyd.

```

Now assume that the user is using a PDA for interacting with the SPA, with dialogue adaption enabled; the example dialogue above would become:

```

User  Is there any new mail from Wayne?
SPA   You have one new message from Wayne Wobcke.
      The message is more than thirty lines, should I
      show you just the summary?
User  Yes please.
SPA   Displays the summary of message from Wayne Wobcke.
      SPA learns to display only the summary
      if the message length is more than 30 lines.
User  Find all messages about meetings in the Inbox.
SPA   SPA has learned in the past to show only the first few
      messages if there are too many messages in the result.
      The message headers should also be sorted by sender.
      I am showing the first ten of twenty messages
      sorted by sender.
      Displays the first 10 message headers sorted by sender.
User  Show me the one from John.
SPA   Here is the summary of the message from John Lloyd.
      SPA shows user the summary of the message according
      the user’s preferences that have been learned so far.

```

Note from the dialogue above how the dialogue manager has learned to adapt its responses according to the user’s preferences. When the user requests the summary of the message from Wayne, the system has learned to show only the summary if the message body is more than thirty lines. Hence later on in the dialogue, the system immediately shows the summary of the e-mail message from John Lloyd because its length is also more than thirty lines. A typical example of a decision list representing the user’s preferences that results in such adaptive behaviour is as follows:

```

if numOfEmail is 0 then
  if plan is ReturnResponse then True
else if numOfEmail is 1 and length is greater than 30 then
  if plan is ReturnSummaryContent then True
  else if numOfEmail is 1 and priority is HIGH then False
else
  if numOfEmail is less than 15 then
    if plan is ReturnMessageSortedBySender then True
    else if plan is ReturnSubList then True

```

3. CONCLUSION

We have proposed a way of incorporating learning into a BDI agent architecture that allows the agent to learn the appropriate plan to select in a given context. The technique can be applied to plan-based dialogue systems to achieve dialogue adaptation, which is particularly useful in dialogue-based applications for mobile devices, where the interaction should be tailored to suit the current conversational context, the physical environment and the user’s device and preferences.

4. ACKNOWLEDGEMENTS

This work was funded by the CRC for Smart Internet Technology. We would like to thank Agent Oriented Software Pty. Ltd. for the use of their JACK agents platform. Many thanks to Kee Siong Ng and Joshua Cole for supplying Alkemy and helping us resolve technical problems arising in the integration of Alkemy and the SPA.

5. REFERENCES

- [1] A. Guerra-Hernández, A. E. Fallah-Seghrouchni and H. Soldano. Learning in BDI Multi-agent Systems. In *CLIMA IV*, pages 218–233. Springer-Verlag, Berlin, 2004.
- [2] N. Howden, R. Rönquist, A. Hodgson and A. Lucas. JACK Intelligent AgentsTM – Summary of an Agent Infrastructure. Paper presented at the *Second International Workshop on Infrastructure for Agents, MAS, and Scalable MAS*, 2001.
- [3] K. Jokinen, A. Kerminen, M. Kaipainen, T. Jauhiainen, G. Wilcock, M. Turunen, J. Hakulinen, J. Kuusisto and K. Lagus. Adaptive Dialogue Systems - Interaction with Interact. In *Proceedings of the 3rd SIGdial Workshop on Discourse and Dialogue*, 2002.
- [4] J. W. Lloyd. *Logic for Learning: Learning Comprehensible Theories from Structured Data*. Springer-Verlag, Berlin, 2003.
- [5] A. Nguyen and W. Wobcke. An Agent-Based Approach to Dialogue Management in Personal Assistants. In *Proceedings of the 2005 International Conference on Intelligent User Interfaces*, pages 137–144, 2005.
- [6] T. Phung, M. Winikoff and L. Padgham. Learning within the BDI Framework: An Empirical Analysis. In R. Khosla, R. J. Howlett and L. C. Jain, editors, *Knowledge-Based Intelligent Information and Engineering Systems, Part III*, pages 282–288. Springer-Verlag, Berlin, 2005.
- [7] C. A. Thompson, M. H. Göker and P. Langley. A Personalized System for Conversational Recommendations. *Journal of Artificial Intelligence Research*, 21:393–428, 2004.
- [8] M. A. Walker. An Application of Reinforcement Learning to Dialogue Strategy Selection in a Spoken Dialogue System for Email. *Journal of Artificial Intelligence Research*, 12:387–416, 2000.
- [9] W. Wobcke, V. Ho, A. Nguyen and A. Krzywicki. A BDI Agent Architecture for Dialogue Modelling and Coordination in a Smart Personal Assistant. In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 323–329, 2005.